Section 4 Worksheet

Part 1: Interpreting SQL and Relational Data

For each SQL query, find
1) what the SQL statement is querying for (a short description) and
2) an equivalent relational algebra (RA) expression/tree

A. (Midterm 12AU)

Clinic(cid, name, street, state)
Equipment(eid, type, model)
Assignment(cid, eid)

Finds the count of clinics that do not have a fridge (of model 1004) assigned to it.

1)
SELECT COUNT(*)
FROM Clinic AS C
WHERE NOT EXISTS (SELECT *
FROM Assignment AS A, Equipment AS E
WHERE C.cid = A.cid AND
A.eid = E.eid AND
E.type = 'Fridge' AND
E.model = 1004);

2)
B. (Midterm 15AU)

**Item**(oid, category, price)
**Gift**(pid, rid, oid) -- pid gifts oid to rid

Select O1.category, max(abs(O1.price - O2.price))
FROM Gift AS G1, Gift AS G2, Item AS O1, Item AS O2
WHERE G1.pid = G2.rid AND
  G2.pid = G1.rid AND
  O1.oid = G1.oid AND
  O2.oid = G2.oid AND
  O1.category = O2.category
GROUP BY O1.category
HAVING count(*) > 5;

1) Finds item categories that have been mutually gifted over 5 times and the corresponding maximum price difference between mutually exchanged items (of said category).

2)
Part 1. Datalog Practice
Consider a graph of colored vertices and undirected edges where the vertices can be red, green, blue. In particular, you have the relations

\[
\begin{align*}
\text{Vertex}(x, \text{color}) \\
\text{Edge}(x, y)
\end{align*}
\]

The Edge relation is symmetric in that if \((x, y)\) is in Edge, then \((y, x)\) is in Edge.
Your goal is to write a datalog program to answer each of the following questions.

1. Find all green vertices.
   \[
   \text{GreenV}(x) := \text{Vertex}(x, 'green')
   \]

2. Find all pairs of blue vertices connected by one edge.
   \[
   \text{BluePairs}(x, y) := \text{Vertex}(x, 'blue'), \text{Vertex}(y, 'blue'), \text{Edge}(x, y)
   \]

3. Find all triangles where all the vertices are the same color. Output the three vertices and their color.
   \[
   \text{Triangle}(x, y, z, c) := \text{Vertex}(x, c), \text{Vertex}(y, c), \text{Vertex}(z, c), \\
   \quad \text{Edge}(x, y), \text{Edge}(y, z), \text{Edge}(z, x)
   \]

4. Find all vertices that don’t have any neighbors.
   \[
   \text{LonelyV}(x) := \neg \text{Edge}(x, _)
   \]
   \[
   \text{LonelyV}(x) := \text{Vertex}(x, _), \neg \text{Edge}(x, _)
   \]
   \[
   \text{CORRECT ANSWER (SAFE)}
   \]
   \[
   \text{OnlyX}(x) := \text{Edge}(x, _)
   \]
   \[
   \text{LonelyV}(x) := \text{Vertex}(x, _), \neg \text{OnlyX}(x)
   \]

5. Find all vertices such that they only have red neighbors.
   \[
   \text{BlueV}(x) := \text{Vertex}(x, _), \text{Edge}(x, y), \text{Vertex}(y, 'blue')
   \]
   \[
   \text{GreenV}(x) := \text{Vertex}(x, _), \text{Edge}(x, y), \text{Vertex}(y, 'green')
   \]
   \[
   \text{RedV}(x) := \text{Vertex}(x, _), \neg \text{BlueV}(x), \neg \text{GreenV}(x)
   \]

6. Find all vertices such that they only have neighbors with the same color. Return the vertex and color.
   \[
   \text{SameColor}(x, y, a) := \text{Vertex}(x, a), \text{Vertex}(y, a)
   \]
   \[
   \text{NotSameNeigh}(x) := \text{Vertex}(x, _), \text{Edge}(x, y), \text{Edge}(x, z), \neg \text{SameColor}(y, z)
   \]
   \[
   \text{OnlySameNeigh}(x, a) := \text{Vertex}(x, a), \neg \text{NotSameNeigh}(x)
   \]
   \[
   \text{OR}
   \]
   \[
   \text{Neigh}(x, y, a) := \text{Edge}(x, y), \text{Vertex}(y, a)
   \]
   \[
   \text{DifferentNeigh}(x) := \text{Neigh}(x, y, a), \text{Neigh}(x, z, b), a \neq b
   \]
   \[
   \text{OnlySameNeigh}(x, a) := \text{Vertex}(x, a), \neg \text{DifferentNeigh}(x)
   \]

7. For some vertex v, find all vertexes connected to v by blue vertexes (this one requires recursion).
   \[
   \text{ConnectedTo}(x) := \text{Vertex}(x, 'blue'), \text{Edge}(x, 'v')
   \]
   \[
   \text{ConnectedTo}(x) := \text{Vertex}(x, 'blue'), \text{Edge}(x, y), \text{ConnectedTo}(y)
   \]