The following problems use these table definitions:

```sql
CREATE TABLE Person  
  ( pid  INT PRIMARY KEY, -- person ID 
    name VARCHAR(100)); -- person name 

CREATE TABLE Email  
  ( eid     INT PRIMARY KEY,       -- email ID 
    pidFrom INT REFERENCES Person, -- email sender 
    length  INT);                  -- email char length 

CREATE TABLE EmailTo  
  ( eid   INT REFERENCES Email,   -- email ID 
    pidTo INT REFERENCES Person,  -- email recipient 
    PRIMARY KEY (eid, pidTo));
```

Relational algebra operators:

<table>
<thead>
<tr>
<th>Operator</th>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Union</td>
<td>⋃</td>
<td>Union</td>
</tr>
<tr>
<td>Difference</td>
<td>−</td>
<td>Difference</td>
</tr>
<tr>
<td>Selection</td>
<td>σ</td>
<td>Selection</td>
</tr>
<tr>
<td>Projection</td>
<td>π</td>
<td>Projection</td>
</tr>
<tr>
<td>Join</td>
<td>⨝</td>
<td>Join</td>
</tr>
<tr>
<td>Rename</td>
<td>ρ</td>
<td>Rename</td>
</tr>
<tr>
<td>Duplicate elimination</td>
<td>δ</td>
<td>Duplicate elimination</td>
</tr>
<tr>
<td>Grouping and aggregation</td>
<td>γ</td>
<td>Grouping and aggregation</td>
</tr>
<tr>
<td>Sorting</td>
<td>τ</td>
<td>Sorting</td>
</tr>
</tbody>
</table>

A witnessing problem: List the pid of people who wrote the longest emails to themselves and the length of the emails.

```sql
SELECT E1.pidFrom, MAX(E2.length) 
FROM Email E1, EmailTo T1, Email E2, EmailTo T2 
WHERE E1.eid = T1.eid AND 
  T1.pidTo = E1.pidFrom AND 
  E2.eid = T2.eid AND 
  T2.pidTo = E2.pidFrom 
GROUP BY E1.pidFrom, E1.length 
HAVING E1.length = MAX(E2.length);
```

Draw the RA tree for the query
The following problems use these table definitions:

CREATE TABLE Person  
    (pid  INT PRIMARY KEY, -- person ID    
     name VARCHAR(100)); -- person name

CREATE TABLE Email  
    (eid     INT PRIMARY KEY,       -- email ID     
     pidFrom INT REFERENCES Person, -- email sender     
     length  INT);                  -- email char length

CREATE TABLE EmailTo  
    (eid   INT REFERENCES Email,   -- email ID   
     pidTo INT REFERENCES Person,  -- email recipient
     PRIMARY KEY (eid, pidTo));

Relational algebra operators:

Union ∪  Difference −  Selection σ  Projection π  Join ⨝
Rename ρ  Duplicate elimination δ  Grouping and aggregation γ  Sorting τ

A subquery problem: Find all emails where all of the recipients are named Alice.

SELECT E1.eid  
FROM Email E1  
WHERE NOT EXISTS (SELECT *  
    FROM EmailTo E2, Person P  
    WHERE E1.eid = E2.eid AND  
    E2.pidTo = P.pid AND  
    P.name != 'Alice');

Write the uncorrelated version of the query

Draw the RA tree for the uncorrelated version of the query