CSE 344 Introduction to Data Management

Section 5: Relational Algebra, Relational Calculus, Datalog

TA: Aloka Krishnan (alokak@uw.edu)

Homework

- Homework 3
  - Due tonight at 11.59pm
- Homework 4
  - Based on RA, RC, Datalog
  - Due Thursday, 13th February 2013, at 11:59pm

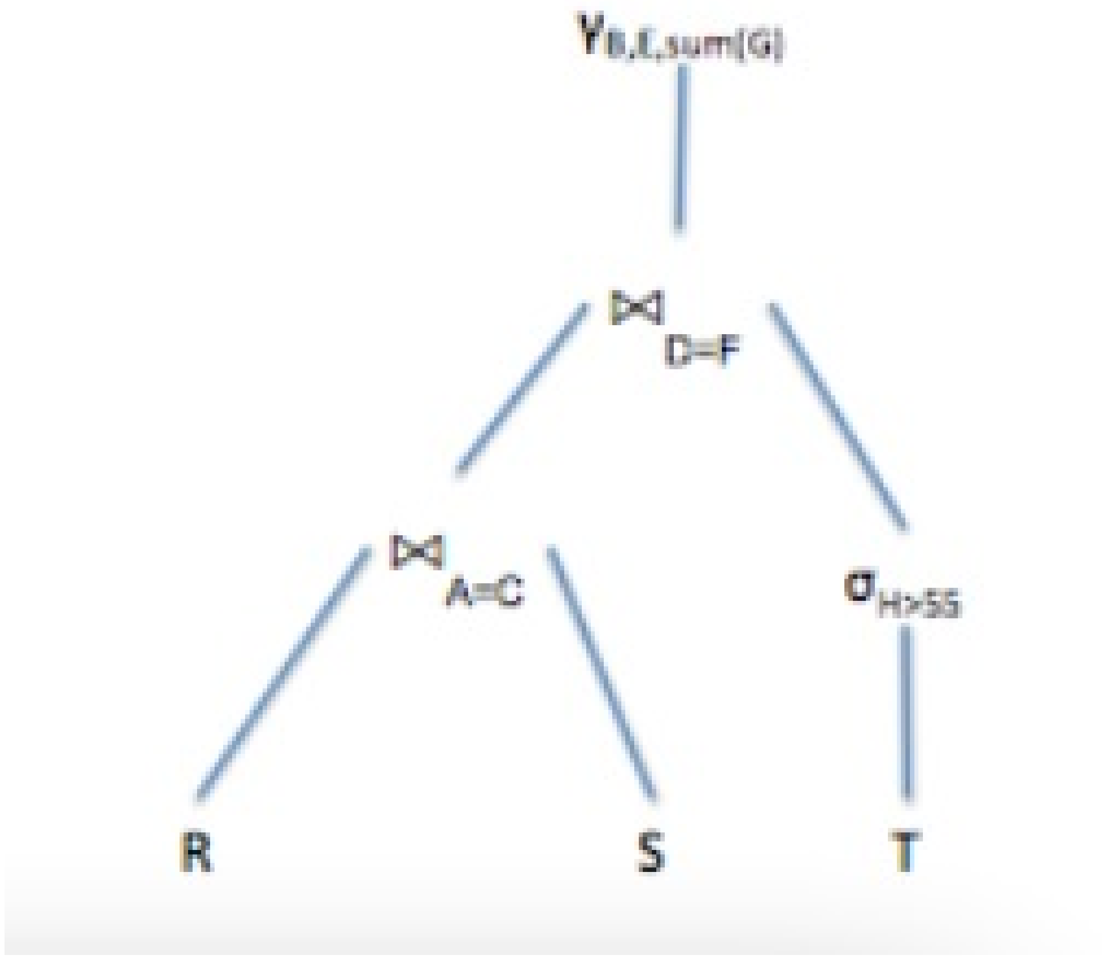
Question 1

Schema: R(A,B), S(C,D,E), T(F,G,H)

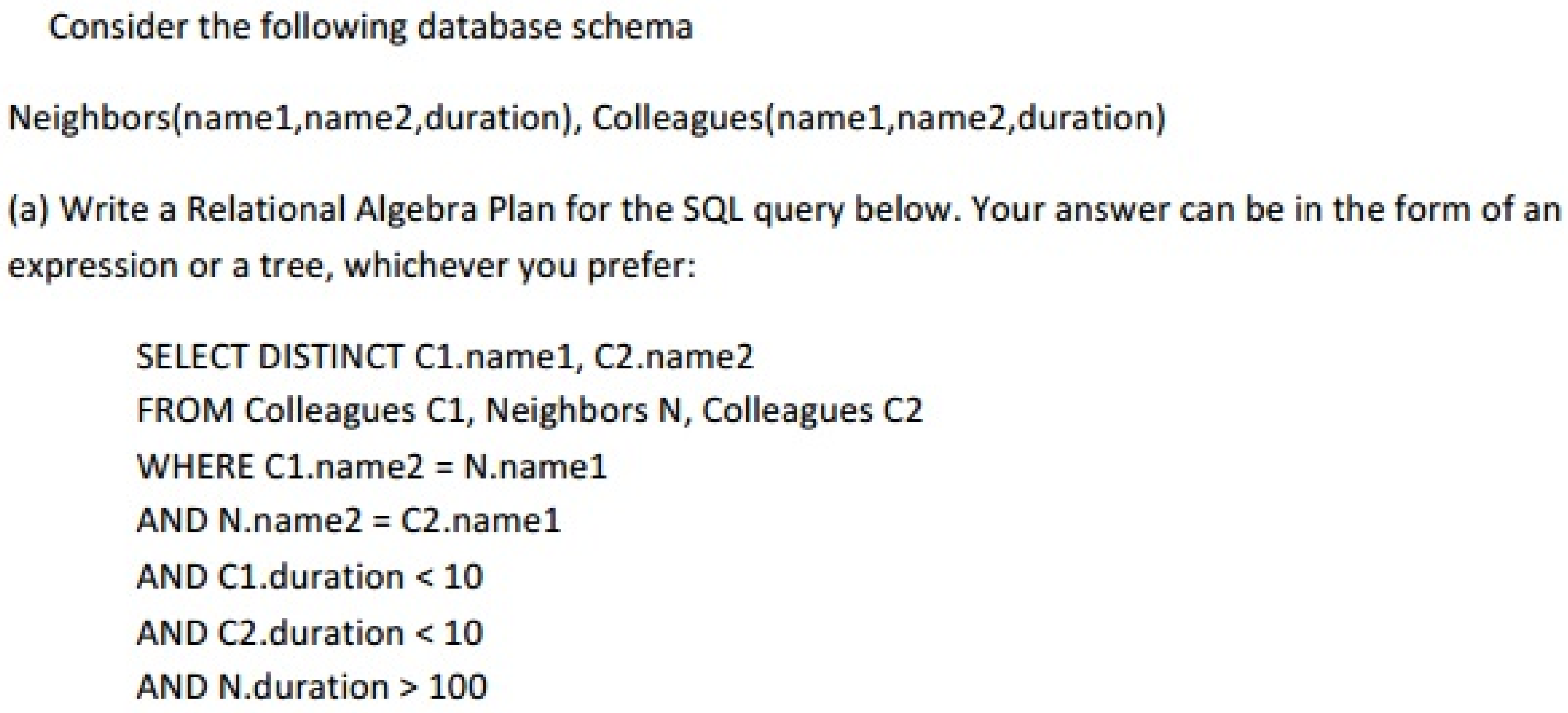
Write a Relational Algebra Plan for the SQL query below. Your answer should be a tree representing the relational algebra plan.

```
SELECT R.B, S.E, sum(T.G)
FROM R, S, T
WHERE R.A = S.C AND S.D = T.F and T.H > 55
GROUP BY R.B, S.E
```

Solution 1



Question 2 (a)



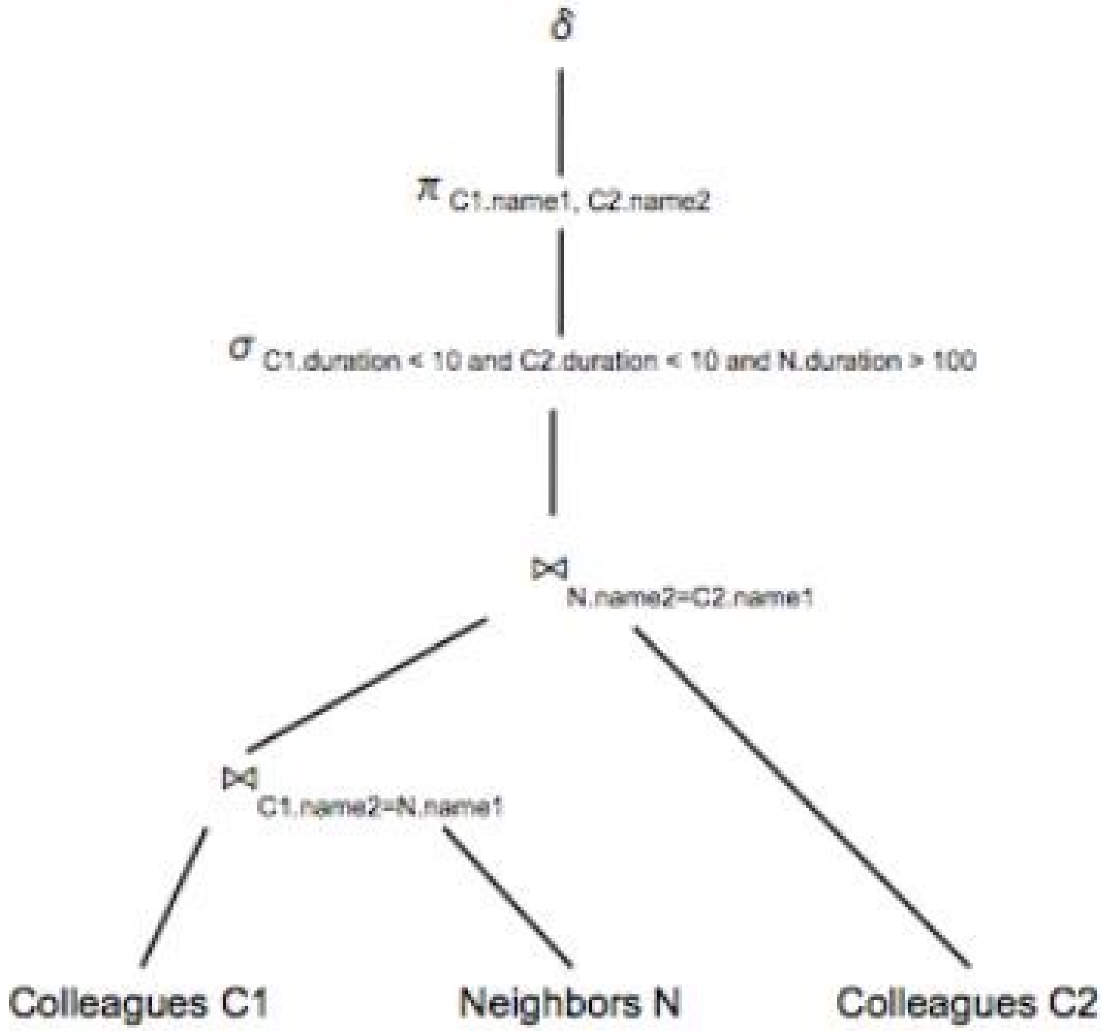
Consider the following database schema

Neighbors(name1,name2,duration), Colleagues(name1,name2,duration)

(a) Write a Relational Algebra Plan for the SQL query below. Your answer can be in the form of an expression or a tree, whichever you prefer:

```
SELECT DISTINCT C1.name1, C2.name2
FROM Colleagues C1, Neighbors N, Colleagues C2
WHERE C1.name2 = N.name1
AND N.name2 = C2.name1
AND C1.duration < 10
AND C2.duration < 10
AND N.duration > 100
```

Solution 2 (a)


δ
π C1.name1, C2.name2
σ C1.duration < 10 and C2.duration < 10 and N.duration > 100
⋈ N.name2=C2.name1
⋈ C1.name2=N.name1
Colleagues C1
Neighbors N
Colleagues C2

Question 2 (b)

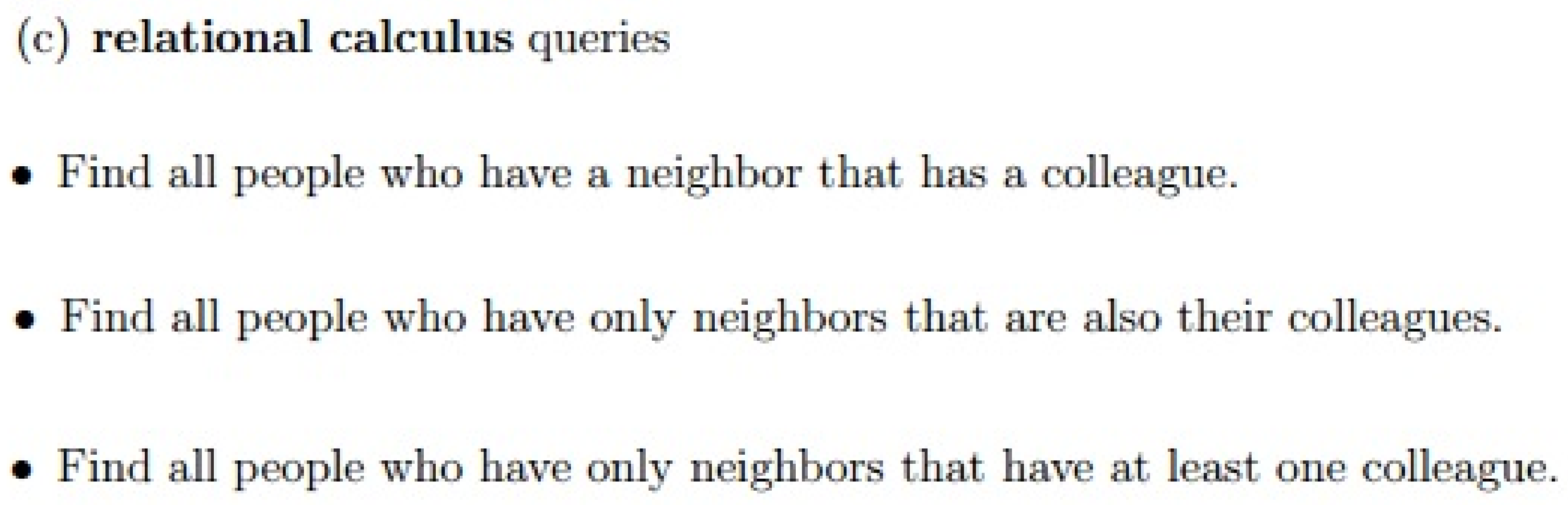
Write a Datalog query that returns all neighbors who do not have any colleagues in common

Solution 2 (b)

NonAnswers(n1,n2):- Neighbors(n1, n2, -), Colleagues(n1, c, -), Colleagues(n2, c, -)

A(n1, n2):- Neighbors(n1, n2, -), NOT NonAnswers(n1,n2)

Question 2 (c)



(c) **relational calculus** queries

- Find all people who have a neighbor that has a colleague.
- Find all people who have only neighbors that are also their colleagues.
- Find all people who have only neighbors that have at least one colleague.

Solution 2 (c)

- Find all people who have a neighbor that has a colleague.

$$A(x) = \exists y.\exists z.Neighbors(x, y, -) \wedge Colleagues(y, z, -)$$

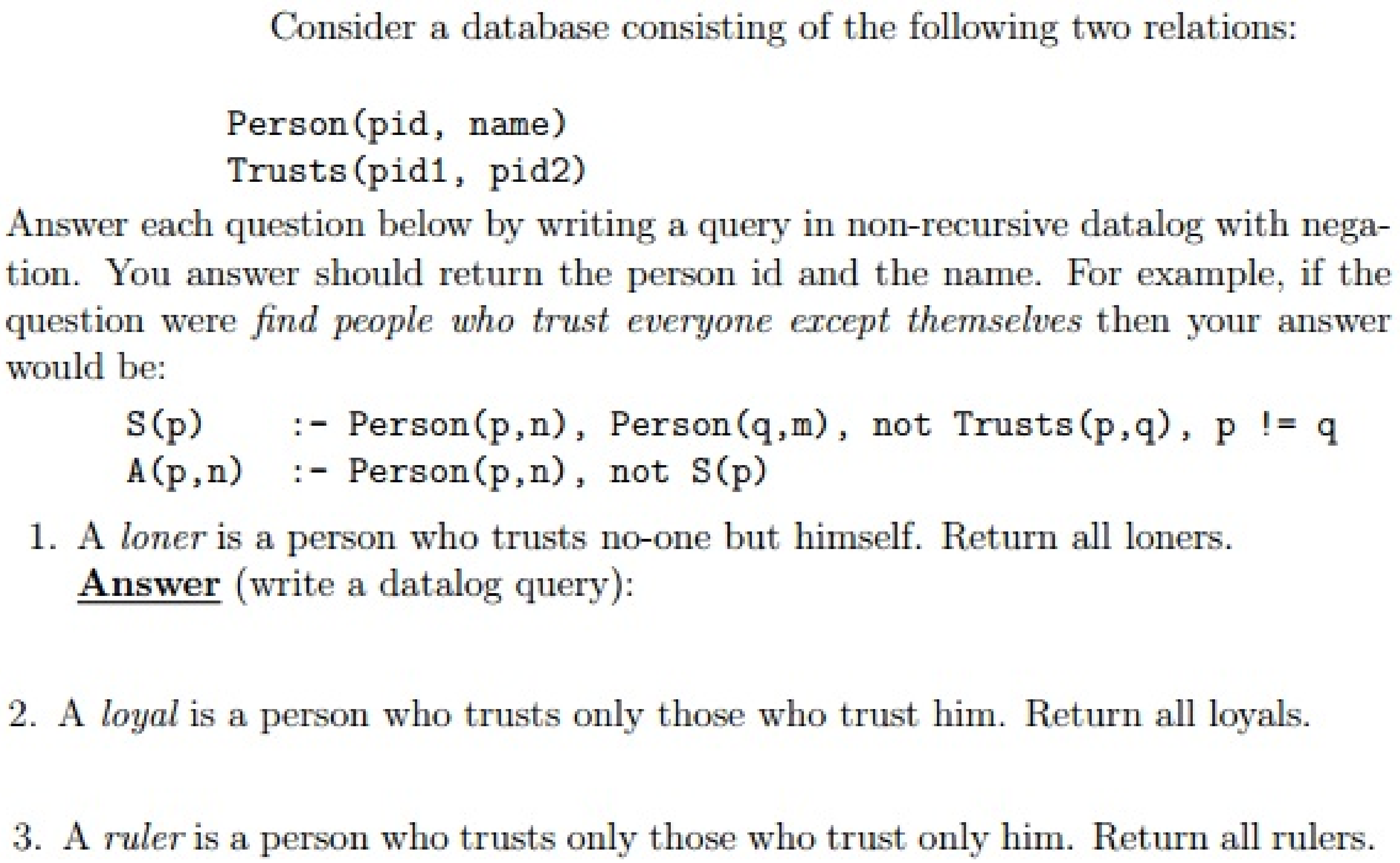
- Find all people who have only neighbors that are also their colleagues.

$$A(x) = Neighbors(x, -, -) \wedge (\forall y.Neighbors(x, y, -) \Rightarrow Colleagues(x, y, -))$$

- Find all people who have only neighbors that have at least one colleague.

$$A(x) = Neighbors(x, -, -) \wedge (\forall y.Neighbors(x, y, -) \Rightarrow \exists z.Colleagues(y, z, -))$$

Question 3



Consider a database consisting of the following two relations:

```
Person(pid, name)
Trusts(pid1, pid2)
```

Answer each question below by writing a query in non-recursive datalog with negation. You answer should return the person id and the name. For example, if the question were *find people who trust everyone except themselves* then your answer would be:

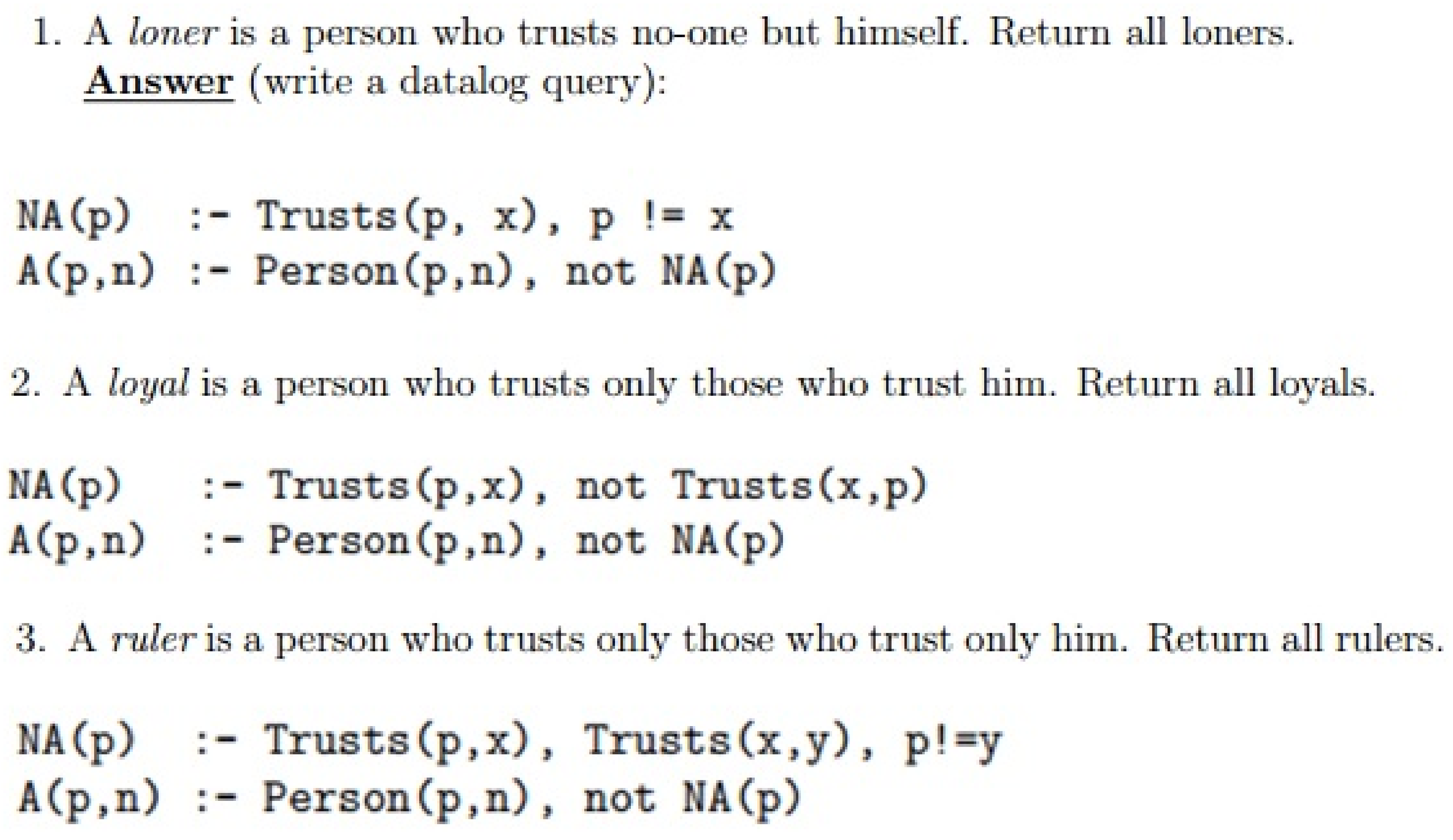
```
S(p)    :- Person(p,n), Person(q,m), not Trusts(p,q), p != q
A(p,n)  :- Person(p,n), not S(p)
```

1. A *loner* is a person who trusts no-one but himself. Return all loners.
   Answer (write a datalog query):

2. A *loyal* is a person who trusts only those who trust him. Return all loyals.

3. A *ruler* is a person who trusts only those who trust only him. Return all rulers.

Solution 3



1. A *loner* is a person who trusts no-one but himself. Return all loners.
   Answer (write a datalog query):

```
NA(p)  :- Trusts(p, x), p != x
A(p,n) :- Person(p,n), not NA(p)
```

2. A *loyal* is a person who trusts only those who trust him. Return all loyals.

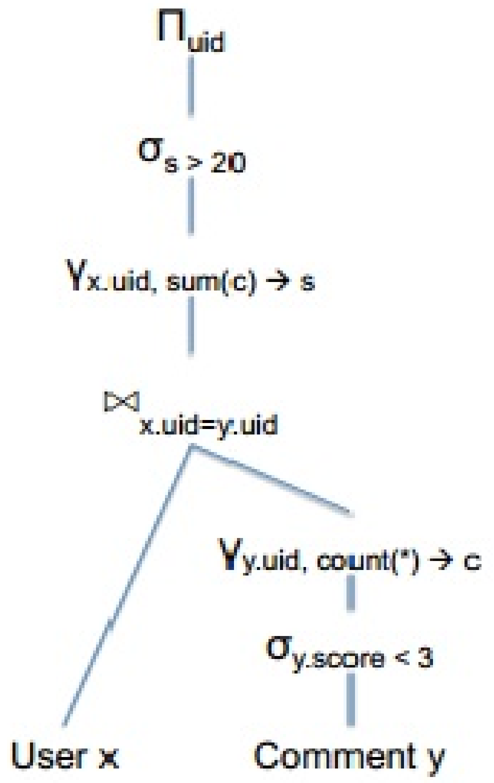
```
NA(p)   :- Trusts(p,x), not Trusts(x,p)
A(p,n)  :- Person(p,n), not NA(p)
```

3. A *ruler* is a person who trusts only those who trust only him. Return all rulers.

```
NA(p)  :- Trusts(p,x), Trusts(x,y), p!=y
A(p,n) :- Person(p,n), not NA(p)
```

Question 4

Consider the Relational Algebra expression below:



Write an equivalent SQL query *without* using any subqueries.

Solution 4

```
select x.uid
from Users x, Comment y
where x.uid = y.uid and y.score < 3
group by x.uid
having count(*) > 20
```