Database schema. For the first two problems we will use the following schema:

**Customer**: $ID$, Name, City

where $ID$ is the primary key.

**Account**: $Number$, Branch, CustID, Balance

where $Number$ is the primary key and $CustID$ is a foreign key referencing $Customer$ on $ID$.

**Problem 1.** Write the following queries in SQL:

1. —
2. “ID and name of customers who do **not** own any account.”
3. “ID and name of customers who own an account in **every** branch.”
4. “ID and name of customers who own an account with a balance which is no less than the balance of any other account.”

[ We have seen how to write these queries in relational algebra and calculus in the previous tutorial. Now try to write them directly in SQL. ]

**Problem 2.** Express the following relational algebra query in relational calculus:

$$\text{CUSTOMER} \bowtie \left(\pi_{\text{ID,City}}(\text{CUSTOMER}) \cap \rho_{\text{CustID} \to \text{ID, Branch} \to \text{City}}(\pi_{\text{Branch,CustID}}(\text{ACCOUNT}))\right)$$

[ Use the algorithm we have seen in class. ]

**Problem 3.** Given two relations $R$ and $S$, each over attributes $A, B$ (in this order), express the following relational calculus query in relational algebra:

$$\{x \mid \neg (\forall y \ R(x, y) \to S(x, y)) \land \neg (\exists z \ S(x, z) \land R(z, x))\}$$

[ Use the algorithm we have seen in class. ]

**Problem 4.**

(a) Can we simplify the relational algebra expression obtained in Problem 3 into an equivalent expression that does not mention the active domain? If yes, give such an expression. If no, explain why this is the case.

(b) How would you translate the relational calculus query of Problem 3 if the output tuple were $x, x$ rather than $x$? [ We have not seen this in class; just think of a way of doing it and we’ll discuss it during the tutorial. ]