Relational Algebra Exercises for Tutorial

Solve all queries below using only select, project, Cartesian product, and natural join. Do not use theta-join, set operations, renaming or assignment.

First Schema

Suppliers(sID, sName, address)

Parts(pID, pName, colour)

Catalog(sID, pID, price)

Catalog[sID] ⊆ Suppliers[sID]

Catalog[pID] ⊆ Parts[pID]

Notice:

- In this schema, everywhere we want values to match across relations, the attributes have matching names. And everywhere the attributes have matching names, we want values to match across relations.
- This means that natural join will do exactly what we want in all cases.

Questions:

1. If sID is a key for the Suppliers relation, could it be a key for the Catalog relation?
2. Find the names of all red parts.
3. Find all prices for parts that are red or green. (A part may have different prices from different manufacturers.)
4. Find the sIDs of all suppliers who supply a part that is red or green.
5. Find the sIDs of all suppliers who supply a part that is red *and* green.
6. Find the names of all suppliers who supply a part that is red or green.

Second Schema

Employees(number, name, age, salary)

Supervises(boss, employee)

Supervises[boss] $\subseteq$ Employees[number]

Supervises[employee] $\subseteq$ Employees[number]

Notice:

- In this schema, wherever we want values to match across relations, the attributes **do not** have matching names. This means that natural join will not force things to match up as we'd like.
- In fact, since there are no attribute names in common across the two relations, natural join is no different from Cartesian product.
- We are forced to use selection to enforce the necessary matching.

Questions:

1. What does it say about our domain that employee is a key for Supervises?
2. Does the schema allow for an employee with no boss?
3. How would the world have to be different if boss were a key for Supervises?
4. How would the world have to be different if both boss and employee together were a key for Supervises?

Third Schema

This schema is for a salon. Services could be things like “haircut” or “manicure”.

Clients(CID, name, phone)

Staff(SID, name)

Appointments(CID, date, time, service, SID)

Appointments[CID] ⊆ Clients[CID]

Appointments[SID] ⊆ Staff[SID]

Notice:

- In this schema, everywhere we want values to match across relations, the attributes have matching names. But there are also attributes with matching names whose values we do not want to match across relations.
- In those cases, that natural join will get rid of many tuples that we need, so we must use Cartesian product and make any necessary matching happen using select. (Unless we can remove the problem attributes first.)

Questions:

1. Find the appointment time and client name of all appointments for staff member Giuliano on Feb14. (Assume that you can compare a date value to “Feb 14” using “=”). At each step, use projection to pare down to only the attributes you need.