Database Management Systems, Aug-Dec 2023

Solution sheet, 1 September 2023

Problem 1 Consider the following relation schema from the university database discussed in the lectures.

`instructor(ID,name,dept_name,salary)`

Write relational algebra queries for the following.

1. Find all faculty members from Physics who earn more than *at least one* faculty member from Comp.Sci.

   Solution

   - Natural self-join of `instructor` with renaming
   - Check that first instructor is from Physics, second is from Comp.Sci and salary of first instructor is higher than that of second instructor
   - Project onto first instructor

   $$\pi_{\texttt{t1.ID}}(\sigma_{\substack{\texttt{t1.dept\_name} = \texttt{'Physics'}\wedge \\ \texttt{t2.dept\_name} = \texttt{'Comp.Sci'}\wedge \\ \texttt{t1.salary} > \texttt{t2.salary}}}(\rho_{\texttt{t1}}(\texttt{instructor}) \bowtie \rho_{\texttt{t2}}(\texttt{instructor})))$$

2. Find all faculty members from Physics who earn more than *every* faculty member from Comp.Sci.

   Solution

   - Find all Physics faculty who earn less than equal to some Comp.Sci faculty
   - Take set difference between first relation and this relation

   $$\pi_{\texttt{t1.ID}}(\sigma_{\substack{\texttt{t1.dept\_name} = \texttt{'Physics'}\wedge \\ \texttt{t2.dept\_name} = \texttt{'Comp.Sci'}\wedge \\ \texttt{t1.salary} > \texttt{t2.salary}}}(\rho_{\texttt{t1}}(\texttt{instructor}) \bowtie \rho_{\texttt{t2}}(\texttt{instructor})))$$
   $$\setminus$$
   $$\pi_{\texttt{t1.ID}}(\sigma_{\substack{\texttt{t1.dept\_name} = \texttt{'Physics'}\wedge \\ \texttt{t2.dept\_name} = \texttt{'Comp.Sci'}\wedge \\ \texttt{t1.salary} \leq \texttt{t2.salary}}}(\rho_{\texttt{t1}}(\texttt{instructor}) \bowtie \rho_{\texttt{t2}}(\texttt{instructor})))$$

3. Find the faculty member(s) with the minimum salary.

   Solution

   - Natural self-join of `instructor` with renaming
   - Check that salary of first instructor is higher than that of second instructor
   - Instructor with minimum salary never appears as first ID in the resulting table
   - Project onto first instructor
   - Take set difference between instructor and this relation

   $$\texttt{instructor} \setminus \pi_{\texttt{t1.ID}}(\sigma_{\texttt{t1.salary}>\texttt{t2.salary}}(\rho_{\texttt{t1}}(\texttt{instructor}) \bowtie \rho_{\texttt{t2}}(\texttt{instructor})))$$

Problem 2 Consider the following relation schema describing a family tree.

```
family(ID,name,gender)
relation(ID1,ID2,relationship)
```

Make the following assumptions:

- In `family`, `gender` takes values `M` or `F`
- In `relation`
  - The Fields `ID1` and `ID2` refer to entries in `ID` from `family`
  - `relationship` takes values `parent` or `spouse`

- The interpretation of a tuple `(id1,id2,parent)` is that `id1` is the parent of `id2`.

Write relational algebra queries for the following.

1. Compute the relation `sibling(ID1,ID2)` — `ID1` is a brother/sister of `ID2`

   Do this for the following intepretations of sibling.

   - `ID1` and `ID2` have at least one parent in common
   - `ID1` and `ID2` have both parents in common

   Solution

   One parent in common:

   - Take natural self join of `relation`, with renaming
   - Check that `ID1` is the same in both copies, `ID2` is different across the two copies, `relationship` is `parent` in both copies.

$$\pi_{\texttt{t1.ID,t2,ID}}(\sigma_{\substack{\texttt{t1.ID1} = \texttt{t2.ID1} \wedge \\ \texttt{t1.ID2} \neq \texttt{t2.ID2} \wedge \\ \texttt{t1.relationship} = \text{'parent'} \wedge \\ \texttt{t2.relationship} = \text{'parent'}}} (\rho_{\texttt{t1}}(\texttt{relationship}) \bowtie \rho_{\texttt{t2}}(\texttt{relationship})))$$

   Both parents in common:

   - Cartesian product of four copies of `relation` with renaming
     - Nested pairwise natural joins
   - First pair checks one parent, second pair checks other parent
   - `t1.ID1 = t2.ID1`, `t3.ID1 = t4.ID1`, `t1.ID1` $\neq$ `t3.ID1`
   - `t1.ID2 = t3.ID2`, `t2.ID2 = t4.ID2`, `t1.ID2` $\neq$ `t2.ID2`
   - `ti.relationship = t2.relationship = t3.relationship = t4.relationship = 'parent'`

$$\pi_{\texttt{t1.ID,t2,ID}}(\sigma_{\substack{\texttt{t1.ID1} = \texttt{t2.ID1} \wedge \\ \texttt{t3.ID1} = \texttt{t4.ID1} \wedge \\ \texttt{t1.ID1} \neq \texttt{t3.ID1} \wedge \\ \texttt{t1.ID2} = \texttt{t3.ID2} \wedge \\ \texttt{t2.ID2} = \texttt{t4.ID2} \wedge \\ \texttt{t1.ID2} \neq \texttt{t2.ID2} \wedge \\ \texttt{t1.relationship} = \text{'parent'} \wedge \\ \texttt{t2.relationship} = \text{'parent'} \wedge \\ \texttt{t3.relationship} = \text{'parent'} \wedge \\ \texttt{t4.relationship} = \text{'parent'}}} (\rho_{\texttt{t1}}(\texttt{relationship}) \bowtie (\rho_{\texttt{t2}}(\texttt{relationship}) \bowtie (\rho_{\texttt{t3}}(\texttt{relationship}) \bowtie \rho_{\texttt{t4}}(\texttt{relationship})))))$$

2. Compute the relation `sister(ID1,ID2)` — `ID1` is a sister of `ID2` with both intepretations of sister, as above.

   Solution

   - Same as above but need to check that gender of `ID1` is `F`

$$\sigma_{\substack{\texttt{gender} = \texttt{'F'} \wedge \\ \texttt{familymember.ID} = \texttt{R.ID1}}} (\texttt{family} \times \texttt{R})$$

   where `R` is the relation computed by the earlier queries.

3. Compute `grandparent(ID1,ID2)` — `ID1` is grandparent of `ID2`

   - Take natural join of `relation` with renaming
   - Check that `t1.ID1` and `t2.ID1` are different
   - Check that `t1.ID2 = t2.ID1`
   - Check that `t1.relationship = t2.relationship = 'Parent'`

$$\pi_{\texttt{t1.ID1},\texttt{t2.ID2}}(\sigma_{\substack{\texttt{t1.ID1} \neq \texttt{t2.ID1} \wedge \\ \texttt{t1.ID2} = \texttt{t2.ID1} \wedge \\ \texttt{t1.relationship} = \text{'parent'} \wedge \\ \texttt{t2.relationship} = \text{'parent'}}} (\rho_{\texttt{t1}}(\texttt{relationship}) \bowtie \rho_{\texttt{t2}}(\texttt{relationship}))))$$

4. Compute `greatgrandparent(ID1,ID2)` — `ID1` is greatgrandparent of `ID2`
   - Take threeway natural join of `relation` with renaming
   - Check that `t1.ID1`, `t2.ID1`, and `t3.ID1` are all different
   - Check that `t1.ID2 = t2.ID1`, `t2.ID2 = t3.ID1`
   - Check that `t1.relationship = t2.relationship = t3.relationship = 'Parent'`

$$\pi_{\texttt{t1.ID1},\texttt{t3.ID2}}(\sigma_{\substack{\texttt{t1.ID1} \neq \texttt{t2.ID1} \wedge \\ \texttt{t2.ID1} \neq \texttt{t3.ID1} \wedge \\ \texttt{t1.ID1} \neq \texttt{t3.ID1} \wedge \\ \texttt{t1.ID2} = \texttt{t2.ID1} \wedge \\ \texttt{t2.ID2} = \texttt{t3.ID1} \wedge \\ \texttt{t1.relationship} = \text{'parent'} \wedge \\ \texttt{t2.relationship} = \text{'parent'} \wedge \\ \texttt{t3.relationship} = \text{'parent'}}} (\rho_{\texttt{t1}}(\texttt{relationship}) \bowtie (\rho_{\texttt{t2}}(\texttt{relationship}) \bowtie \rho_{\texttt{t3}}(\texttt{relationship}))))$$

5. Can you compute `ancestor(ID1,ID2)` in general?
   - As we have seen with `grandparent` and `greatgrandparent`, for each additional level of ancestor, we have to do one extra join
   - A general `ancestor` relation has an unbounded number of levels, so will require an unbounded natural join
   - Can compute `ancestor` upto a fixed distance, but not in general

---