Translation from SQL into the relational algebra

Consider the following relational schema:

- Student(snum, sname, major, level, age)
- Class(name, meets_at, room, fid)
- Enrolled(snum, cname)
- Faculty(fid, fname, deptid)

Task

Translate the following SQL-query into an expression of the relational algebra.

```
SELECT S.sname
FROM Student S
WHERE S.snum NOT IN (SELECT E.snum FROM Enrolled E)
```

Translation from SQL into the relational algebra

Solution

```
SELECT S.sname
FROM Student S
WHERE S.snum NOT IN (SELECT E.snum FROM Enrolled E)
```

First, the query is normalized to a form in which only `EXISTS` and `NOT EXISTS` occur:

```
SELECT S.sname
FROM Student S
WHERE NOT EXISTS
  (SELECT E.snum FROM Enrolled E WHERE E.snum = S.snum)
```

Translation from SQL into the relational algebra

Solution (continued)

Translation of the subquery

```
(SELECT E.snum FROM Enrolled E WHERE E.snum = S.snum)
```

gives us the expression

$$E_1 := \boldsymbol{\pi}_{\texttt{S.snum,S.sname,S.major,S.level,S.age,E.snum}}\, \boldsymbol{\sigma}_{\texttt{E.snum=S.snum}} \\ (\boldsymbol{\rho}_{\texttt{E}}(\texttt{Enrolled}) \times \boldsymbol{\rho}_{\texttt{S}}(\texttt{Student}))$$

Translation of the from-where part without subqueries of the whole query gives:

$$E_2 := \boldsymbol{\rho}_{\texttt{S}}(\texttt{Student})$$

The decorrelation of the subquery gives:

$$E_3 := E_2 \;\overline{\bowtie}\; \boldsymbol{\pi}_{\texttt{S.snum,S.sname,S.major,S.level,S.age}}(E_1)$$

Finally, we translate the remaining projection:

$$\boldsymbol{\pi}_{\texttt{S.sname}}(E_3)$$

Translation from SQL into the relational algebra

Solution (continued)

Written in full:

$$\boldsymbol{\pi}_{\mathtt{S.sname}}(\boldsymbol{\rho}_{\mathtt{S}}(\mathtt{Student}) \mathbin{\overline{\bowtie}} \boldsymbol{\pi}_{\mathtt{S.snum,S.sname,S.major,S.level,S.age}} \boldsymbol{\sigma}_{\mathtt{E.snum=S.snum}} (\boldsymbol{\rho}_{\mathtt{E}}(\mathtt{Enrolled}) \times \boldsymbol{\rho}_{\mathtt{S}}(\mathtt{Student}))))$$

(Notice that we merged the two consecutive projections of E_1)

Translation from SQL into the relational algebra

Consider again the following relational schema:

- Student(snum, sname, major, level, age)
- Class(name, meets_at, room, fid)
- Enrolled(snum, cname)
- Faculty(fid, fname, deptid)

Task

Translate the following SQL-query into an expression of the relational algebra.

```
SELECT C.name  FROM Class C
WHERE C.room = 'R128'
OR C.name IN (SELECT E.cname FROM Enrolled E
              GROUP BY E.cname HAVING COUNT(*) >= 5)
```

Translation from SQL into the relational algebra

Solution

```
SELECT C.name  FROM Class C
WHERE C.room = 'R128'
OR C.name IN (SELECT E.cname FROM Enrolled E
              GROUP BY E.cname HAVING COUNT(*) >= 5)
```

First, the query is normalized to a form in which only `EXISTS` and `NOT EXISTS` occur:

```
SELECT C.name FROM Class C
WHERE C.room = 'R128'
OR EXISTS (SELECT E.cname FROM Enrolled E
           WHERE E.cname = C.name
           GROUP BY E.cname HAVING COUNT(*) >= 5)
```

Translation from SQL into the relational algebra

Solution (continued)

We then convert it into a union of queries whose selection list only contains conjunctions:

```
( SELECT C.name FROM Class C
  WHERE C.room = 'R128' )
UNION
( SELECT C.name FROM Class C
  WHERE EXISTS
   (SELECT E.cname FROM Enrolled E
    WHERE E.cname = C.name
    GROUP BY E.cname HAVING COUNT(*) >= 5)
)
```

Translation from SQL into the relational algebra

Solution (continued)

We first translate

```
SELECT C.name FROM Class C
WHERE C.room = 'R128'
```

$$E_1 := \boldsymbol{\pi}_{\texttt{C.name}}\, \boldsymbol{\sigma}_{\texttt{C.room}='\texttt{R128}'}\, \boldsymbol{\rho}_{\texttt{C}}(\texttt{Class})$$

For the other part of the union, we consider the subquery first

```
SELECT E.cname FROM Enrolled E
WHERE E.cname = C.name
GROUP BY E.cname HAVING COUNT(*) >= 5
```

$$E_2 := \boldsymbol{\pi}_{\texttt{E.cname},\texttt{C.name},\texttt{C.meets\_at},\texttt{C.room},\texttt{C.fid}}\, \boldsymbol{\sigma}_{\texttt{COUNT}(*)>=5}\, \boldsymbol{\gamma}_{\texttt{E.cname},\texttt{COUNT}(*),\texttt{C.name},\texttt{C.meets\_at},\texttt{C.room},\texttt{C.fid}}\, \boldsymbol{\sigma}_{\texttt{E.cname}=\texttt{C.cname}}(\boldsymbol{\rho}_{\texttt{E}}(\texttt{Enrolled}) \times \boldsymbol{\rho}_{\texttt{C}}(\texttt{Class}))$$

Translation from SQL into the relational algebra

Solution (continued)

The translation of the from-where part of the surrounding query without its subqueries is:

$$E_3 := \boldsymbol{\rho}_{\texttt{C}}(\texttt{Class})$$

The decorrelation of the subquery gives:

$$E_4 := \hat{E}_3 \bowtie \boldsymbol{\pi}_{\texttt{C.name,C.meets\_at,C.room,C.fid}}(E_2)$$

Notice that $\hat{E}_3$ is totally empty! The full translation is therefore

$$E_1 \cup \boldsymbol{\pi}_{\texttt{C.name}}(E_4)$$

Written in full:

$$\boldsymbol{\pi}_{\texttt{C.name}}\, \boldsymbol{\sigma}_{\texttt{C.room}='\texttt{R128}'}\, \boldsymbol{\rho}_{\texttt{C}}(\texttt{Class})$$
$$\cup\, \boldsymbol{\pi}_{\texttt{C.cname}}\, \boldsymbol{\sigma}_{\texttt{COUNT}(*)>=5}\, \boldsymbol{\gamma}_{\texttt{E.cname,COUNT}(*)\texttt{,C.name,C.meets\_at,C.room,C.fid}}$$
$$\boldsymbol{\sigma}_{\texttt{E.cname=C.cname}}(\boldsymbol{\rho}_{\texttt{E}}(\texttt{Enrolled}) \times \boldsymbol{\rho}_{\texttt{C}}(\texttt{Class})).$$

Again, we have merged successive projections.

Translation from SQL into the relational algebra

Consider again the following relational schema:

- Student(snum, sname, major, level, age)
- Class(name, meets_at, room, fid)
- Enrolled(snum, cname)
- Faculty(fid, fname, deptid)

Task

Translate the following SQL-query into an expression of the relational algebra.

```
SELECT F.fname
FROM Faculty F
WHERE 5 >  (SELECT COUNT(E.snum)
            FROM Class C, Enrolled E
            WHERE C.name = E.cname AND
                  C.fid = F.fid)
```

Translation from SQL into the relational algebra

Solution

First, the query is normalized to a form in which only EXISTS and NOT EXISTS occur:

```
SELECT F.fname
FROM Faculty F
WHERE EXISTS (SELECT COUNT(E.snum)
              FROM Class C, Enrolled E
              WHERE C.name = E.cname AND
                    C.fid = F.fid
              HAVING COUNT(E.snum) < 5)
```

The translation of the subquery gives:

$$E_1 := \boldsymbol{\pi}_{\texttt{COUNT(E.snum),F.fid,F.fname,F.deptid}}\, \boldsymbol{\sigma}_{\texttt{COUNT(E.snum)}<5}\, \boldsymbol{\gamma}_{\texttt{COUNT(E.snum),F.fid,F.fname,F.deptid}}\, \boldsymbol{\sigma}_{\texttt{C.name=E.cname}\wedge\texttt{C.fid=F.fid}}\, (\boldsymbol{\rho}_{\texttt{C}}(\texttt{Class}) \times \boldsymbol{\rho}_{\texttt{E}}(\texttt{Enrolled}) \times \boldsymbol{\rho}_{\texttt{F}}(\texttt{Faculty}))$$

Translation from SQL into the relational algebra

Solution (continued)

The translation of the whole query without subquery and projection is

$$E_2 = \boldsymbol{\rho}_{\mathtt{F}}(\mathtt{Faculty})$$

The decorrelation of the subquery gives:

$$E_3 = \hat{E}_2 \bowtie \boldsymbol{\pi}_{\mathtt{F.fid,F.fname,F.deptid}}(E_1)$$

Notice that $\hat{E}_2$ is empty! The final query is therefore:

$$E_4 = \boldsymbol{\pi}_{\mathtt{F.fname}}(E_3)$$

After merging the projections, we get:

$$\boldsymbol{\pi}_{\mathtt{F.fname}}\, \boldsymbol{\sigma}_{\mathtt{COUNT(E.snum)<5}}$$
$$\boldsymbol{\gamma}_{\mathtt{COUNT(E.snum),F.fid,F.fname,F.deptid}}\, \boldsymbol{\sigma}_{\mathtt{C.name=E.cname \wedge C.fid=F.fid}}$$
$$(\boldsymbol{\rho}_{\mathtt{C}}(\mathtt{Class}) \times \boldsymbol{\rho}_{\mathtt{E}}(\mathtt{Enrolled}) \times \boldsymbol{\rho}_{\mathtt{F}}(\mathtt{Faculty}))$$

Which is really not equivalent to the original SQL query!

Translation from SQL into the relational algebra

Solution (continued)

The translation is not equivalent to the original SQL query! Indeed, faculty members who teach no class will not occur in the output of E_4, while they will occur in the output of the original SQL query.

This phenomenon is known as the `COUNT` bug. This bug occurs only when we have subqueries that use `COUNT` without `GROUP BY`. We can solve this as follows:

$$\boldsymbol{\pi}_{\texttt{F.fname}}\, \boldsymbol{\sigma}_{\texttt{COUNT(E.snum)}<5}\boldsymbol{\gamma}_{\texttt{COUNT(E.snum),F.fid,F.fname,F.deptid}}\, \boldsymbol{\sigma}_{\texttt{C.name=E.cname}} \left((\boldsymbol{\rho}_{\texttt{C}}(\texttt{Class}) \times \boldsymbol{\rho}_{\texttt{E}}(\texttt{Enrolled})) \overset{\text{O}}{\underset{\texttt{C.fid=F.fid}}{\bowtie_{\text{R}}}} \boldsymbol{\rho}_{\texttt{F}}(\texttt{Faculty})\right)$$

Notice that we can only take the outer join with the context relation(s).