Operations on Relations-Division

pm jat @ daiict

Division operation

- Following are example queries that require division
  - **`SupplyParts`** Schema: Suppliers that supply all parts
  - Company Schema: List employees who work on all projects controlled by dno=4.
- Division is typically required when you want to find out entities that are interacting with all entities of a set.
- It is not supported by SQL implementations .. can be represented using other operations … bit complex

Division- definition

Given two relations; r(x,y) , s(y)

r DIV s gives all distinct values of x from r that are associated with all values of y in s.

Division operation – example#1

Given following two relations;

supplies(sid,pid)

parts(pid)

supplies DIV parts
gives us SIDs that supply all PIDs?

sid integer	pid integer
101	1
102	1
101	3
103	2
102	2
102	3
102	4
102	5

pid integer
1
2
3
4
5

Division operation – example#2

- Given following two relations;

  works(ssn,pno)

  proj(pno)

works DIV proj
gives you SSNs that work on all PNOs?

ssn numeric(9,0)	pno smallint
101	2
101	3
101	10
101	20
101	1
102	30
102	20
103	30
103	10
104	3
105	1
105	2
106	10
106	30
107	1
107	2
108	20

pno smallint
10
30

Division – computation

- Let us take relations shown here **works** (left) and **proj** (right) and compute following-

$$r1 \leftarrow \Pi_{ssn}(works) \times proj = ?$$

$$r2 \leftarrow r1 - works = ?$$

$$r2x \leftarrow \Pi_{ssn}(r2) = ?$$

$$r3 \leftarrow \Pi_{ssn}(works) - r2x = ?$$

ssn numeric(9,0)	pno smallint
101	2
101	3
101	10
101	20
101	1
102	30
102	20
103	30
103	10
104	3
105	1
105	2
106	10
106	30
107	1
107	2
108	20

pno smallint
10
30

Division – computation

- **R(x,y) div S(y)** gives you distinct values of x from R that are associated with every value of y in S.
- SQL does not support DIVISION operations; needs to computed using other operation. Can be derived as following[Elmasri/Navathe]
  - Find out all possible combinations of **S(y)** with **R(x)** by computing **R(x)** x **S(y)**, say **r1**
  - Subtract actual **R(x,y)** from **r1**, say **r2**,
  - **x** in **r2** are those that are not associated with every value in **S(y)**; therefore **R(x)-r2(x)** gives us **x** that are associated with all values in **S**

$$R \div S = \pi_x(R) - \pi_x((\pi_x(R) \times S) - R)$$

Division computation

All possible combinations
r1 ← π_x(R) x S
x values with "incomplete combinations",
r2x ← π_x(r1-R)
and result -
π_x(R)-r2x

$$\pi_x(R) - \pi_x((\pi_x(R) \times S) - R)$$

sid integer	pid integer
101	1
102	1
101	3
103	2
102	2
102	3
102	4
102	5

sid integer
101
102
103

pid integer
1
2
3
4
5

sid integer	pid integer
101	1
101	2
101	3
101	4
101	5
102	1
102	2
102	3
102	4
102	5
103	1
103	2
103	3
103	4
103	5

sid integer	pid integer
101	2
101	4
101	5
103	1
103	3
103	4
103	5

sid integer
101
103

sid integer
102

Computation of Division

- It should be visible that division may not be directly performed on full relations; relations needs to be brought down to fit them in division form.
- For example, actual relations for query “Suppliers that supply all parts”. Given relations are Supplies(sid,pid,cost), and Parts(pid,pname,color); necessary projections are needed; this query solution is expressed as-

$$\Pi_{sid,pid}(Supplies) \ div \ \pi_{pid}(Parts)$$

pid	pname	color
1	PART-1	RED
2	PART-2	GREEN
3	PART-3	RED
4	PART-4	BLUE
5	PART-5	GREEN

sid	pid	cost
101	1	100
102	1	120
101	3	160
103	2	210
102	2	220
102	3	150
102	4	400
102	5	500

SQL Solution (Strategy) - 1

R(x,y) DIV S(y) be expressed as

SELECT x that are
NOT IN
All - Actual

```
SELECT x FROM R
WHERE x NOT IN (
  SELECT x FROM (
     ( All possible; i.e. S x π_x(R) )
          MINUS
     ( Actual R )
   )
);
```

Strategy#1 applied

"Suppliers that supply all parts"

```
SELECT sid FROM Suppliers
WHERE sid NOT IN (
  SELECT sid FROM (
     ( All possible sid, pid combinations)
       MINUS
     ( Actual sid, pid pairs from Supplies )
);
```

Strategy#1 applied

"Suppliers that supply all parts"

```
SELECT * FROM suppliers
WHERE sid not in (
SELECT sid FROM (
  (SELECT sid, pid FROM (select pid from
  parts) as p cross join (select distinct sid
  from supplies) as sp)
  EXCEPT
  (SELECT sid, pid FROM supplies)
 ) AS r
);
```

SQL Solution (Strategy) - 2

R(x,y) DIV S(y) be expressed as

```
SELECT x FROM R
WHERE empty-set (
   ( all y, i.e. S )
     MINUS
   ( y that are associate with the x)
);
```

Strategy#2 applied

"Suppliers that supply all parts"

```
SELECT suppliers
WHERE empty-set (
   ( All Parts )
    MINUS
   ( Parts Supplied by the Supplier )
);
```

Strategy#2 applied

"Suppliers that supply all parts"

```
SELECT * FROM suppliers as s
WHERE NOT EXISTS (
    ( SELECT p.pid FROM parts as p )
     EXCEPT
   (SELECT sp.pid FROM supplies sp WHERE sp.sid = s.sid )
);
```

For division correlated query seems simpler to write but may expensive to execute

Division Example #2

List employees who work on all projects controlled by dno=4

- PNOs controlled by dno = 4
  p4 ← $\pi_{PNO}(\sigma_{DNO=4}(PROJECTS))$
- Have ESSN, PNO project of WORKS on relation-
  SSN_PNOS(SSN, PNO)← $\pi_{ESSN, PNO}(WORKS\_ON)$
- SSN of employees works on PNOs in p4
  SSN_PNOS div p4

Using Strategy#2

List employees who work on all projects controlled by dno=4

SELECT employee
WHERE empty-set (
    (all PNOs controlled by dno=4, i.e. p4)
      MINUS
    (PNOs on which the employee works)
);

Using Strategy#2

List employees who work on all projects controlled by dno=4

```
SELECT * FROM employee AS e
WHERE NOT EXISTS (
  (SELECT pno FROM project WHERE dno = 4)
   EXCEPT
  (SELECT pno FROM works_on WHERE essn = e.ssn)
);
```

Using Strategy#1

List employees who work on all projects controlled by dno=4

```
SELECT * FROM EMPLOYEE
WHERE ssn NOT IN (
    SELECT essn FROM (
        ( All possible essn, pno combinations)
        MINUS
        ( Actual essn, pno pairs from WORKS_ON )
);
```

Using Strategy#1

List employees who work on all projects controlled by dno=4

```
SELECT * FROM employee AS e
WHERE ssn NOT IN (
    SELECT essn FROM (
    (SELECT essn, pno FROM (select pno from project where dno=4)
    as p cross join (select distinct essn from works_on) as w)
    EXCEPT
    (SELECT essn, pno FROM works_on)
) AS r
);
```

Division Example#3

Students taken all courses that PMJ offered from academic year 2007-08 to 2011-12.

$$r1 \leftarrow \sigma_{iname='PMJ'}(instructor)$$

$$r2 \leftarrow \sigma_{acadyr>=2007 \text{ and } acadyr<=2011}(offers)$$

$$r3 \leftarrow r1 * r2 * registers$$

$$r4 \leftarrow \Pi_{sid,course,acadyear,semester}(r3)$$

$$r5 \leftarrow \Pi_{course,acadyear,semester}(r3)$$

$$result \leftarrow r4 \text{ div } r5$$

Division Example#3

Students taken all courses that PMJ offered from academic year 2007-08 to 2011-12.

[Using Strategy#1]

SELECT Students

WHERE sid NOT IN (

(All possible combination of sid, cno, yr, sem for PMJ and during specified acad-years)

MINUS

(actual combination of sid, cno, yr, sem in registers for PMJ and during specified acad-years)

);

Division Example#3

Students taken all courses that PMJ offered from academic year 2007-08 to 2011-12.

[Using Strategy#1]

```
SELECT * FROM student AS s
WHERE studentid NOT IN (
   SELECT studentid FROM (
      SELECT studentid, courseno, acadyear, semester from
         ((select courseno, acadyear, semester FROM offers NATURAL JOIN instructor
          WHERE instructorname = 'P M Jat' AND acadyear >= 2007 AND  acadyear <= 2011) as co
            CROSS JOIN (select distinct studentid from registers) as sr)
          EXCEPT
            (SELECT studentid, courseno, acadyear, semester FROM
              registers WHERE acadyear >= 2007 AND acadyear <= 2011)
   ) as r
);
```

Division Example#3

Students taken all courses that PMJ offered from academic year 2007-08 to 2011-12.

[Using Strategy#2]

SELECT Students

WHERE empty-set (

(All courses by PMJ and during specified acad-years)

MINUS

(Courses taken by the StudID during specified acad-years)

);

Division Example#3

Students taken all courses that PMJ offered from academic year 2007-08 to 2011-12.

[Using Strategy#2]

```
SELECT * FROM student AS s
WHERE NOT EXISTS (
        (SELECT courseno, acadyear, semester FROM offers NATURAL JOIN
instructor WHERE instructorname = 'P M Jat' AND acadyear >= 2007 AND
acadyear <= 2011)
        EXCEPT
        (SELECT courseno, acadyear, semester FROM registers AS r WHERE
acadyear >= 2007 AND acadyear <= 2011 AND r.studentid=s.studentid)
);
```

More queries requiring DIVISION

- Retrieve the names of employees, who work on all the projects that ‘John Smith’ works
- List supplier who supply all ‘Red’ Parts
- List students who registered for all courses offered for BTech 2011 Autumn Semester