Today's lecture is on the Division Operation of Relational Algebra

DIVISION OPERATION

The DIVISION operation, denoted by ÷, is useful for a special kind of query that sometimes occurs in database applications.
An example is :

Q1. Retrieve the names of employees who work on all the projects that 'John Smith' works on.
  To express this query using the DIVISION operation, we proceed as follows.

   First, retrieve the list of project numbers that 'John Smith' works on in the relation   SMITH_PNOS:
   1.   SMITH ← σFname='John' AND Lname='Smith'(EMPLOYEE)

  (We join the relations works- on and Smith on (Essn=ssn))

   2. SMITH_PNOS ← πPno. (WORKS_ON.   (JOIN symbol) Essn=Ssn) SMITH

   3 . SSN_PNOS ← πEssn, Pno(WORKS_ON)
      This relation denotes that the employee with SSN = Essn is working on project whose number is Pno.

      Finally, apply the DIVISION operation to the two relations, which gives the desired employees' Social Security numbers:

   4. SSNS(Ssn) ← SSN_PNOS ÷ SMITH_PNOS
    5. RESULT ← πFname, Lname(SSNS * EMPLOYEE)

In general, the DIVISION operation is applied to two relations $R(Z) \div S(X)$, where the attributes of R
 are a subset of the attributes of S; that is, $X \subseteq Z$. Let Y be the set of attributes of R that are not attributes of S; that is, $Y = Z - X$ (and hence $Z = X \cup Y$). The result of DIVISION is a relation T(Y) that includes a tuple t if tuples tR appear in R with R[Y]=t,and with R[x] = tS for every tuple in S . This means that ,for a tuple t to appear in the result T of the DIVISION, the values in t must appear in R in combination with **every tuple in S.** Note that in the formulation of the DIVISION operation, the tuples in the denominator relation S restrict the numerator relation R by selecting those tuples in the result that match all values present in the denominator . The following table illustrates a DIVISION operation where $X = \{A\}$, $Y = \{B\}$, and $Z = \{A, B\}$. Notice that the tuples (values) b1 and b4 appear in R in combination with all three tuples in S; that is why they appear in the resulting relation T. All other values of B in R do not appear with all the tuples in S and are not selected: b2 does not appear with a1.

See the diagram below that illustrates the division operator for the above two examples.
We are using the Company database that we have provided you in the practical class exercises.

(a)

SSN_PNOS

Essn	Pno
123456789	1
123456789	2
666884444	3
453453453	1
453453453	2
333445555	2
333445555	3
333445555	10
333445555	20
999887777	30
999887777	10
987987987	10
987987987	30
987654321	30
987654321	20
888665555	20

SMITH_PNOS

Pno
1
2

SSNS

Ssn
123456789
453453453

(b)

R

A	B
a1	b1
a2	b1
a3	b1
a4	b1
a1	b2
a3	b2
a2	b3
a3	b3
a4	b3
a1	b4
a2	b4
a3	b4

S

A
a1
a2
a3

T

B
b1
b4

Now try to solve the following query using the division operator

Q 1. retrieve the names of all employees who work on every project

Specify the following queries on the COMPANYrelational database schema as given to you in the lab exercises ,using the relational operators discussed in this chapter.

a. Retrieve the names of all employees in department 5 who work more than 10 hours per week on the ProductX project.
b. List the names of all employees who have a dependent with the same first name as themselves.
c. Find the names of all employees who are directly supervised by 'Franklin Wong'.
d. For each project, list the project name and the total hours per week (by all employees)
    Spent on that project.

Following is the summary of the different Relational operators that we have studied in this chapter

SELECT	Selects all tuples that satisfy the selection condition from a relation R.	$\sigma_{<\text{selection condition}>}(R)$
PROJECT	Produces a new relation with only some of the attributes of R, and removes duplicate tuples.	$\pi_{<\text{attribute list}>}(R)$
THETA JOIN	Produces all combinations of tuples from R_1 and R_2 that satisfy the join condition.	$R_1 \bowtie_{<\text{join condition}>} R_2$
EQUIJOIN	Produces all the combinations of tuples from R_1 and R_2 that satisfy a join condition with only equality comparisons.	$R_1 \bowtie_{<\text{join condition}>} R_2$, OR $R_1 \bowtie_{(<\text{join attributes 1}>),\ (<\text{join attributes 2}>)} R_2$
NATURAL JOIN	Same as EQUIJOIN except that the join attributes of R_2 are not included in the resulting relation; if the join attributes have the same names, they do not have to be specified at all.	$R_1 *_{<\text{join condition}>} R_2$, OR $R_1 *_{(<\text{join attributes 1}>),\ (<\text{join attributes 2}>)} R_2$ OR $R_1 * R_2$
UNION	Produces a relation that includes all the tuples in R_1 or R_2 or both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cup R_2$
INTERSECTION	Produces a relation that includes all the tuples in both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cap R_2$
DIFFERENCE	Produces a relation that includes all the tuples in R_1 that are not in R_2; R_1 and R_2 must be union compatible.	$R_1 - R_2$
CARTESIAN PRODUCT	Produces a relation that has the attributes of R_1 and R_2 and includes as tuples all possible combinations of tuples from R_1 and R_2.	$R_1 \times R_2$
DIVISION	Produces a relation $R(X)$ that includes all tuples $t[X]$ in $R_1(Z)$ that appear in R_1 in combination with every tuple from $R_2(Y)$, where $Z = X \cup Y$.	$R_1(Z) \div R_2(Y)$