# ST.XAVIER’S COLLEGE

# MAITIGHAR, KATHMANDU

****

**ASSIGNMENT #6**

**Database Management System**

**Submitted By:**

Pratibha Panta

013BSCCSIT028

4th Sem, 2nd Year

**Submitted To:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Er. Sanjay Kumar Yadav

Department of Computer Science

Lecturer

**1. Join**

An JOIN clause is used to combine rows from two or more tables, based on a common field between them.[1]

Consider the following two tables, (a) CUSTOMERS table is as follows:

ID | NAME | AGE | ADDRESS | SALARY |

+----+----------+-----+-----------+----------+

| 1 | Ramesh | 32 | Ahmedabad | 2000.00 |

| 2 | Khilan | 25 | Delhi | 1500.00 |

| 3 | kaushik | 23 | Kota | 2000.00 |

| 4 | Chaitali | 25 | Mumbai | 6500.00 |

| 5 | Hardik | 27 | Bhopal | 8500.00 |

| 6 | Komal | 22 | MP | 4500.00 |

| 7 | Muffy | 24 | Indore | 10000.00

(b) Another table is ORDERS as follows:

OID | DATE | CUSTOMER\_ID | AMOUNT |

+-----+---------------------+-------------+--------+

| 102 | 2009-10-08 00:00:00 | 3 | 3000 |

| 100 | 2009-10-08 00:00:00 | 3 | 1500 |

| 101 | 2009-11-20 00:00:00 | 2 | 1560 |

| 103 | 2008-05-20 00:00:00 | 4 | 2060 |

Now, let us join these two tables in our SELECT statement as follows:

SQL> SELECT ID, NAME, AGE, AMOUNT

FROM CUSTOMERS, ORDERS

WHERE CUSTOMERS.ID = ORDERS.CUSTOMER\_ID;

This would produce the following result:

ID | NAME | AGE | AMOUNT |

+----+----------+-----+--------+

| 3 | kaushik | 23 | 3000 |

| 3 | kaushik | 23 | 1500 |

| 2 | Khilan | 25 | 1560 |

| 4 | Chaitali | 25 | 2060 |

+----+----------+-----+--------+

**1.1 Theta Join**

Theta join combines tuples from different relations provided they satisfy the theta condition. The join condition is denoted by the symbol θ.[2]

Notation

R1 ⋈θ R2

R1 and R2 are relations having attributes (A1, A2, .., An) and (B1, B2,.. ,Bn) such that the attributes don’t have anything in common, that is R1 ∩ R2 = Φ.

Theta join can use all kinds of comparison operators.

|  |  |  |
| --- | --- | --- |
| **Student** | | |
| **SID** | **Name** | **Std** |
| 101 | Alex | 10 |
| 102 | Maria | 11 |

|  |  |
| --- | --- |
| **Subjects** | |
| **Class** | **Subject** |
| 10 | Math |
| 10 | English |
| 11 | Music |
| 11 | Sports |

Student\_Detail −

STUDENT ⋈Student.Std = Subject.Class SUBJECT

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Student\_detail** | | | | |
| **SID** | **Name** | **Std** | **Class** | **Subject** |
| 101 | Alex | 10 | 10 | Math |
| 101 | Alex | 10 | 10 | English |
| 102 | Maria | 11 | 11 | Music |
| 102 | Maria | 11 | 11 | Sports |

**1.2 Natural Join**

A NATURAL JOIN is a JOIN operation that creates an implicit join clause for you based on the common columns in the two tables being joined. Common columns are columns that have the same name in both tables.[3]

The associated tables have one or more pairs of identically named columns. The columns must be the same data type.[1]

A NATURAL JOIN can be an INNER join, a LEFT OUTER join, or a RIGHT OUTER join.

**1.2.1** Right Join: Return all rows from the right table, and the matched rows from the left table. [1]

**1.2.2** Left Join: Return all rows from the left table, and the matched rows from the right table.[1]

**1.2.3** Inner Join: Returns all rows when there is at least one match in BOTH tables.[1]

**2. Rename operation**

The results of relational algebra are also relations but without any name. The rename operation allows us to rename the output relation. 'Rename' operation is denoted with small Greek letter **rho** *ρ*.

Notation − ρ x (E), Where the result of expression E is saved with name of x.[4]

**3. Assignment operator**

The assignment operation (←) provides a convenient way to express complex queries

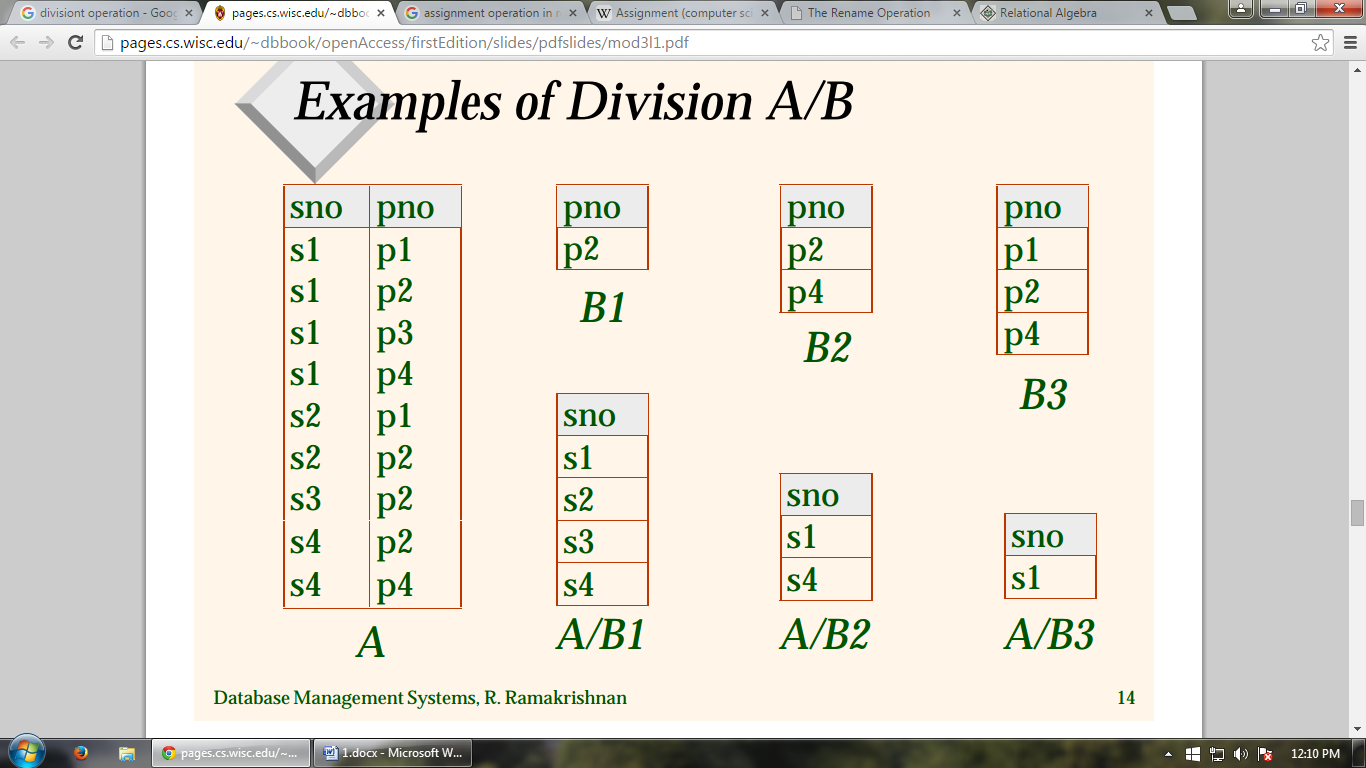
– Write query as a sequential program consisting of a series of assignments followed by an expression whose value is displayed as a result of the query[5]

– Assignment must always be made to a temporary relation variable

**4. Division operation**

Not supported as a primitive operator, but useful for expressing queries.[6]

Division is not essential op; just a useful shorthand. – (Also true of joins, but joins are so common that systems implement joins specially.)



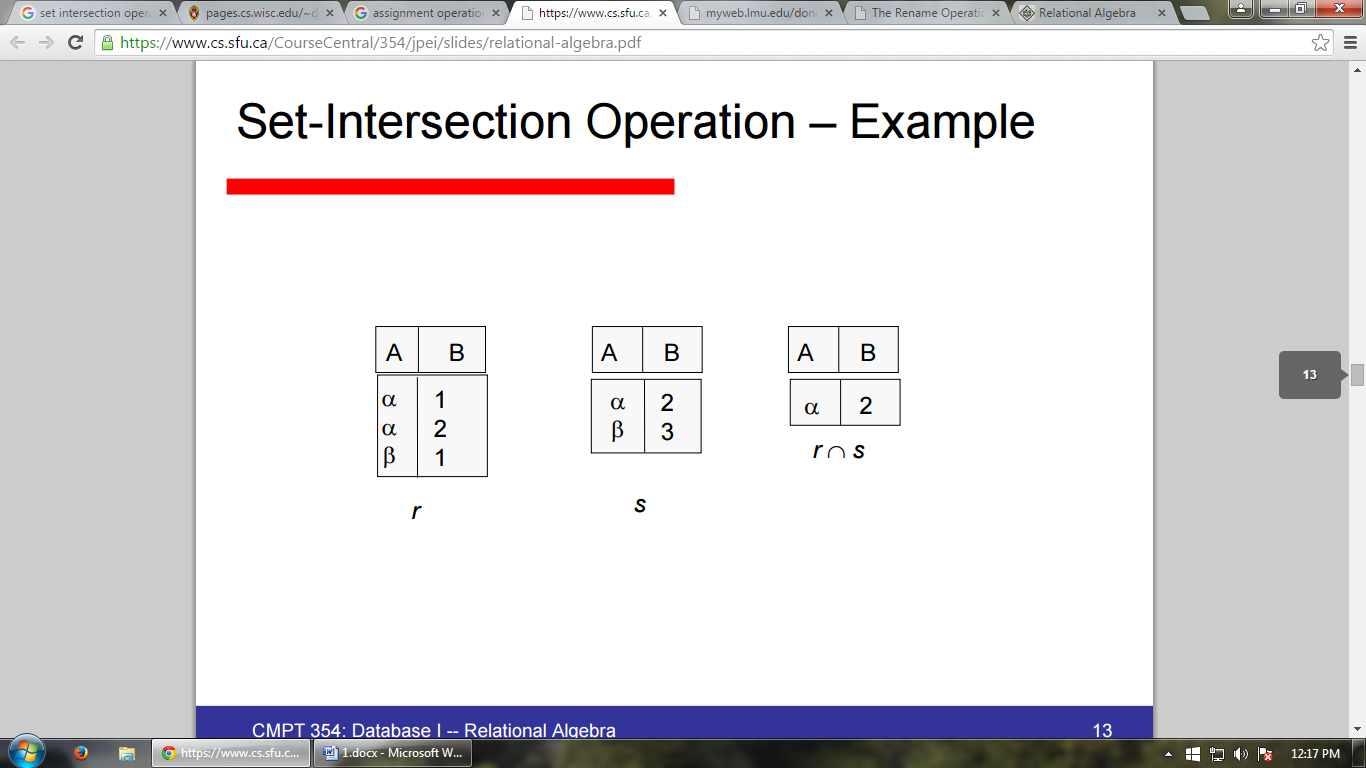
For A/B, compute all x values that are not `disqualified’ by some y value in B.

**5. Additional operations**

“Additional operations” refer to relational algebra operations that can be expressed in terms of the fundamentals — select, project, union, set-difference, cartesian-product, and rename.[7]

**5.1 Self intersection operation**

These operations take two input relations, which must be union-compatible: – Same number of fields. – `Corresponding’ fields have the same type and which are common in both the relations.[5]



**5.2 Natural join operation**

• The natural-join operation is a binary operation on relations r(R) and s(S) that is denoted by the symbol ⋈. [7]

• Intuitively, a natural-join “matches” the tuples of r with the tuples of s based on attributes that are both in r and s.

• If we take the relational schemas R and S as sets of attributes, then we can define “attributes that are in both r and s” as R ∩ S = {A1, A2, . . . , An}.

With that, we can formally define r ⋈s as: r ⋈ s = ΠR ∪ S(σr.A1 = s.A1 ∧ r.A2 = s.A2 ∧ ... ∧ r.An = s.An (r × s))

• Note that R ∪ S removes duplicate attribute names, so r ⋈s will only have one attribute Ak ∀Ak ∈ R ∩ S.

• Natural join is associative — that is, (a ./ b) ./ c = a ./ (b ./ c).

• When r and s do not have any common attributes — i.e., R ∩ S = ∅ — then r ⋈ = r × s.

**Reference**

[1] <http://www.w3schools.com/> (10th September 2015)

[2] <http://www.tutorialspoint.com/dbms/database_joins.htm> (10th September 2015)

[3] <https://docs.oracle.com/javadb/10.8.3.0/ref/rrefsqljnaturaljoin.html> (10th September 2015)

[4] <http://www.tutorialspoint.com/dbms/relational_algebra.htm> (10th September 2015)

[5] <https://www.cs.sfu.ca/CourseCentral/354/jpei/slides/relational-algebra.pdf> (10th September 2015)

[6] <http://pages.cs.wisc.edu/~dbbook/openAccess/firstEdition/slides/pdfslides/mod3l1.pdf> (10th September 2015)

[7] <http://myweb.lmu.edu/dondi/share/db/relational3.pdf> (10th September 2015)