**ST. XAVIER’S COLLEGE**

**(Affiliated to Tribhuwan University)**

**Maitighar, Kathmandu**

**DBMS Theory Assignment#6**

**Submitted By:**

**Dwarika Shiwakoti**

**013BSCCSIT020**

**Submitted To:**

**Er.Sanjaya Kumar Yadav**

**(Lecturer)**

**Department of Computer Science**

**JOIN**

Join is a combination of a Cartesian product followed by a selection process. A Join operation pairs two tuples from different relations, if and only if a given join condition is satisfied. We will briefly describe various join types in the following sections. They are:

**Theta (θ) Join**

Theta join combines tuples from different relations provided they satisfy the theta condition. The

Join condition is denoted by the symbol θ.

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 |

**NATURAL JOIN**

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.

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

If the SELECT statement in which the NATURAL JOIN operation appears has an asterisk (\*) in the select list, the asterisk will be expanded to the following list of columns (in this order):

* All the common columns
* Every column in the first (left) table that is not a common column
* Every column in the second (right) table that is not a common column

An asterisk qualified by a table name (for example, COUNTRIES.\*) will be expanded to every column of that table that is not a common column.

If a common column is referenced without being qualified by a table name, the column reference points to the column in the first (left) table if the join is an INNER JOIN or a LEFT OUTER JOIN. If it is a RIGHT OUTER JOIN, unqualified references to a common column point to the column in the second (right) table.

Syntax

[*TableExpression*](https://docs.oracle.com/javadb/10.8.3.0/ref/rreftableexpression.html#rreftableexpression) NATURAL [ { LEFT | RIGHT } [ OUTER ] | INNER ] JOIN { [TableViewOrFunctionExpression](https://docs.oracle.com/javadb/10.8.3.0/ref/rrefsqlj33215.html#rrefsqlj33215) | ( [*TableExpression*](https://docs.oracle.com/javadb/10.8.3.0/ref/rreftableexpression.html#rreftableexpression) ) }

Examples

If the tables COUNTRIES and CITIES have two common columns named COUNTRY and COUNTRY\_ISO\_CODE, the following two SELECT statements are equivalent:

SELECT \* FROM COUNTRIES NATURAL JOIN CITIES

SELECT \* FROM COUNTRIES JOIN CITIES

USING (COUNTRY, COUNTRY\_ISO\_CODE)

The following example is similar to the one above, but it also preserves unmatched rows from the first (left) table:

SELECT \* FROM COUNTRIES NATURAL LEFT JOIN CITIES

**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.

**The Assignment Operation**

Sometimes it is useful to be able to write a relational algebra expression in parts using a temporary relation variable (as we did with http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1251.gif and http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1261.gif in the division example).

The assignment operation, denoted http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1425.gif, works like assignment in a programming language.

We could rewrite our division definition as

http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_eqnarray516.gif

No extra relation is added to the database, but the relation variable created can be used in subsequent expressions. Assignment to a permanent relation would constitute a modification to the database.

|  |  |  |
| --- | --- | --- |
|  |  |  |

**The Division Operation**

Division, denotedhttp://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1375.gif, is suited to queries that include the phrase ``for all''.

Suppose we want to find all the customers who have an account at all branches located in Brooklyn.

Strategy: think of it as three steps.

We can obtain the names of all branches located in Brooklyn by

http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_displaymath1319.gif

We can also find all *cname, bname* pairs for which the customer has an account by

http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_displaymath1320.gif

Now we need to find all customers who appear in http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1261.gif with every branch name in http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1251.gif.

The divide operation provides exactly those customers:

http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_eqnarray492.gif

which is simply http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1381.gif.

Formally,

Let http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1383.gif and http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1385.gif be relations.

Let http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1387.gif.

The relation http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1409.gif is a relation on scheme http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1405.gif.

A tuple http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1447.gif is in http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1409.gif if for every tuple http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1397.gif in http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1531.gif there is a tuple http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1401.gif in http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1345.gif satisfying both of the following:

http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_eqnarray502.gif

These conditions say that the http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1405.gif portion of a tuple http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1447.gif is in http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1409.gif if and only if there are tuples with the http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1291.gifportion and the http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1335.gif portion in http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1345.gif for every value of the http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1335.gif portion in relation http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1335.gif.

We will look at this explanation in class more closely.

The division operation can be defined in terms of the fundamental operations.

http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_displaymath1321.gif

**Additional operations**

Additional operations are defined in terms of the fundamental operations. They do not add power to the algebra, but are useful to simplify common queries.

There are three types of additional operations. they are:

* Set intersection
* Assignment
* Natural join

**The Set Intersection Operation**

Set intersection is denoted by http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1329.gif, and returns a relation that contains tuples that are in both of its argument relations.

It does not add any power as

http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_displaymath1315.gif

To find all customers having both a loan and an account at the SFU branch, we write

http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_eqnarray430.gif

**The Natural Join Operation**

Often we want to simplify queries on a Cartesian product.

For example, to find all customers having a loan at the bank and the cities in which they live, we need *borrow*  and *customer* relations:

http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_eqnarray439.gif

Our selection predicate obtains only those tuples pertaining to only one *cname*.

This type of operation is very common, so we have the natural join, denoted by a http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1365.gif sign. Natural join combines a Cartesian product and a selection into one operation. It performs a selection forcing equality on those attributes that appear in both relation schemes. Duplicates are removed as in all relation operations.

To illustrate, we can rewrite the previous query as

http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_displaymath1316.gif

We can now make a more formal definition of natural join.

Consider http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1181.gif and http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1335.gif to be sets of attributes.

We denote attributes appearing in both relations by http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1337.gif.

We denote attributes in either or both relations by http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1339.gif.

Consider two relations http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1383.gif and http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1385.gif.

The natural join of http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1345.gif and http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1531.gif, denoted by http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1349.gif is a relation on scheme http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1339.gif.

It is a projection onto http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1339.gif of a selection on http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1355.gif where the predicate requires http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1357.gif for each attribute http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1681.gif in http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1337.gif.

Formally,

http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_displaymath1317.gif

where http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1363.gif

To find the assets and names of all branches which have depositors living in Stamford, we need *customer*, *deposit* and*branch* relations:

http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_eqnarray475.gif

Note that http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1365.gif is associative.

To find all customers who have both an account and a loan at the SFU branch:

http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_displaymath1318.gif

This is equivalent to the set intersection version we wrote earlier. We see now that there can be several ways to write a query in the relational algebra.

If two relations http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1383.gif and http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1385.gif have no attributes in common, then http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1371.gif, and http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter3/_7092_tex2html_wrap1373.gif.