**Join**

We understand the benefits of taking a Cartesian product of two relations, which gives us all the possible tuples that are paired together. But it might not be feasible for us in certain cases to take a Cartesian product where we encounter huge relations with thousands of tuples having a considerable large number of attributes.

**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.

Types of Joins

1. Natural Join
2. Equi Join
3. Outer join

* Left outer Join
* Right outer Join
* Full outer Join

Natural Join (⋈)

Natural join does not use any comparison operator. It does not concatenate the way a Cartesian product does. We can perform a Natural Join only if there is at least one common attribute that exists between two relations. In addition, the attributes must have the same name and domain.

Natural join acts on those matching attributes where the values of attributes in both the relations are same.

Employee

|  |  |  |
| --- | --- | --- |
| Emp.no | Emp.name | Address |
| E1 | Anjit | Kathmandu |
| E2 | Krishna | Pokhara |
| E3 | Anish | Birgunj |
| E4 | Pabitra | Jhapa |

Department

|  |  |  |
| --- | --- | --- |
| Dept.no | Dept.name | Emp.no |
| D1 | BCA | E1 |
| D2 | BBS | E3 |
| D3 | BBM | E4 |

Retrieve the employee’s name who works on the department.

Cross Product

|  |  |  |  |
| --- | --- | --- | --- |
| Emp.no | Emp.name | Dept.no | Emp.no |
| E1 | Anjit | D1 | E1 |
| E1 | Anjit | D2 | E3 |
| E1 | Anjit | D3 | E4 |
| E2 | Krishna | D1 | E1 |
| E2 | Krishna | D2 | E3 |
| E2 | Krishna | D3 | E4 |
| E3 | Anish | D1 | E1 |
| E3 | Anish | D2 | E3 |
| E3 | Anish | D3 | E4 |
| E4 | Pabitra | D1 | E1 |
| E4 | Pabitra | D2 | E3 |
| E4 | Pabitra | D3 | E4 |

∏ Emp.name ( Employee ⋈ Department)

Select emp.name from employee Inner Join on Department where employee.emp.no=

Department.emp.n0;

Select emp.name from employee Natural Join on department;

|  |
| --- |
| Emp.name |
| Anjit |
| Anish |
| Pabitra |

**Outer Join**

Equijoin, and Natural Join are called inner joins. An inner join includes only those tuples with matching attributes and the rest are discarded in the resulting relation. Therefore, we need to use outer joins to include all the tuples from the participating relations in the resulting relation. There are three kinds of outer joins − left outer join, right outer join, and full outer join.

Employee

|  |  |  |
| --- | --- | --- |
| Emp.no | Emp.name | Address |
| E1 | Anjit | Kathmandu |
| E2 | Krishna | Pokhara |
| E3 | Anish | Birgunj |
| E4 | Pabitra | Jhapa |

Department

|  |  |  |
| --- | --- | --- |
| Dept.no | Dept.name | Emp.no |
| D1 | BCA | E1 |
| D2 | BBS | E2 |
| D3 | BBM | E3 |

**Left Outer Join(R Left Outer Join S)**

All the tuples from the Left relation, R, are included in the resulting relation. If there are tuples in R without any matching tuple in the Right relation S, then the S-attributes of the resulting relation are made NULL.

Retrieve the emp.no, emp.name, dept.no, dept.name from the above relations.

∏ Emp.no, Emp.name, Dept.no, Dept.name ( Employee Left Outer Join Department)

|  |  |  |  |
| --- | --- | --- | --- |
| Emp.no | Emp.name | Dept.no | Dept.name |
| E1 | Anjit | D1 | BCA |
| E3 | Anish | D3 | BBM |
| E4 | Pabitra | null | null |
| E2 | Krishna | D2 | BBS |

**Right Outer Join: ( R Right Outer Join S )**

All the tuples from the Right relation, S, are included in the resulting relation. If there are tuples in S without any matching tuple in R, then the R-attributes of resulting relation are made NULL.

Retrieve the emp.no, emp.name, dept.no, dept.name from the above relations.

∏ Emp.no, Emp.name, Dept.no, Dept.name ( Employee Right Outer Join Department)

|  |  |  |  |
| --- | --- | --- | --- |
| Emp.no | Emp.name | Dept.no | Dept.name |
| E1 | Anjit | D1 | BCA |
| E2 | Krishna | D2 | BBS |
| E3 | Anish | D3 | BBM |

**Full Outer Join: ( R Full Outer Join S)**

All the tuples from both participating relations are included in the resulting relation. If there are no matching tuples for both relations, their respective unmatched attributes are made NULL.

|  |  |  |  |
| --- | --- | --- | --- |
| Emp.no | Emp.name | Dept.no | Dept.name |
| E1 | Anjit | D1 | BCA |
| E3 | Anish | D3 | BBM |
| E4 | Pabitra |  |  |
| E3 | Anish | D3 | BBM |

**Equi Join**

It is also known as an inner join. It is the most common join. It is based on matched data as per the equality condition. The equi join uses the comparison operator(=).

.

1. Consider the following database, where primary keys are underlined

student (SID, SName, semester)

studies (SID, CID)

course (CID, CName, CCode)

Construct the following relational algebra and SQL queries for this database.

1. Find the names of all students in third semester.

Projection sname ( select semester = “third” (student))

1. Find the names of all courses studied by Ram.

∏ CName ( Select SName =”Ram”(course ⋈ student ⋈ studies)