# WEEK 3: READING MATERIAL 2

## Types of Joins

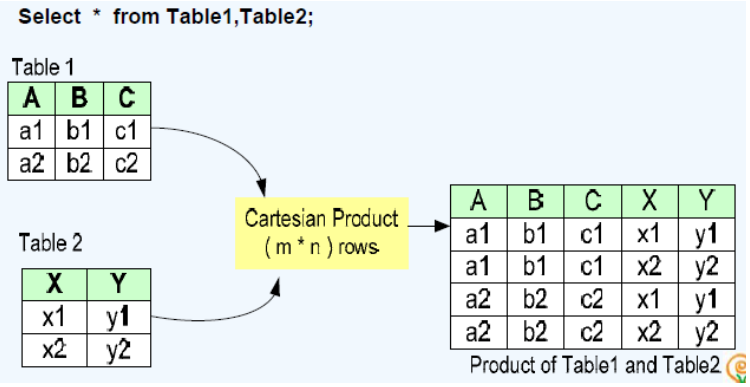
Following are the type of join.

- Cartesian Product
- Inner join
  - Equi join
  - Non-equi join
- Outer join
  - Left-outer join
  - Right-outer join
- Self join

## Cartesian product

A Cartesian join occurs when data is selected from two or more tables and there is no common relation specified in the WHERE clause. If we do not specify a join condition for the tables listed in the FROM clause, Oracle joins each row from the first table to every row in the second table. In Cartesian join each row from the first table is combined with all rows from the second table.

- The working of Cartesian product is illustrated below with example database.



```
Select * from Table1,Table2;
```

Table 1

| A | B | C |
|---|---|---|
| a1 | b1 | c1 |
| a2 | b2 | c2 |

Table 2

| X | Y |
|---|---|
| x1 | y1 |
| x2 | y2 |

| A | B | C | X | Y |
|---|---|---|---|---|
| a1 | b1 | c1 | x1 | y1 |
| a1 | b1 | c1 | x2 | y2 |
| a2 | b2 | c2 | x1 | y1 |
| a2 | b2 | c2 | x2 | y2 |

Product of Table1 and Table2

If the first table has three rows and the second table has four rows, the result will have 12 rows. Suppose we add another table with two rows without specifying a join condition; the result will have 24 rows. We should avoid Cartesian joins; for the most part, they happen when there are many tables in the FROM clause and developers forget to include the join condition.

**Example:**

```
SQL>Select ename,dname FROM emp, dept;
```

The above query displays employee name and department name from EMP and DEPT tables. Because no WHERE clause has been specified , all rows( 14 rows) from EMP table are joined with all (4 rows) in the DEPT table, there by generating 56 rows in the output as shown below:

EMP (14 rows)

| **EMPNO** | **ENAME** | **DEPTNO** |
|---|---|---|
| 7839 | KING | 10 |
| 7698 | BLAKE | 30 |
| ……. | ……. | ….. |
| 7934 | MILLER | 10 |

DEPT (4 rows)

| **DEPTNO** | **DNAME** | **LOC** |
|---|---|---|
| 10 | ACCOUNTING | NEW YORK |
| 20 | RESEARCH | DALLAS |
| 30 | SALES | CHICAGO |
| 40 | OPERATIONS | BOSTON |

| **ENAME** | **DNAME** |
|---|---|
| KING | ACCOUNTING |
| BLAKE | ACCOUNTING |
| …… | ……… |
| KING | RESEARCH |
| BLAKE | RESEARCH |
| …… | …….. |

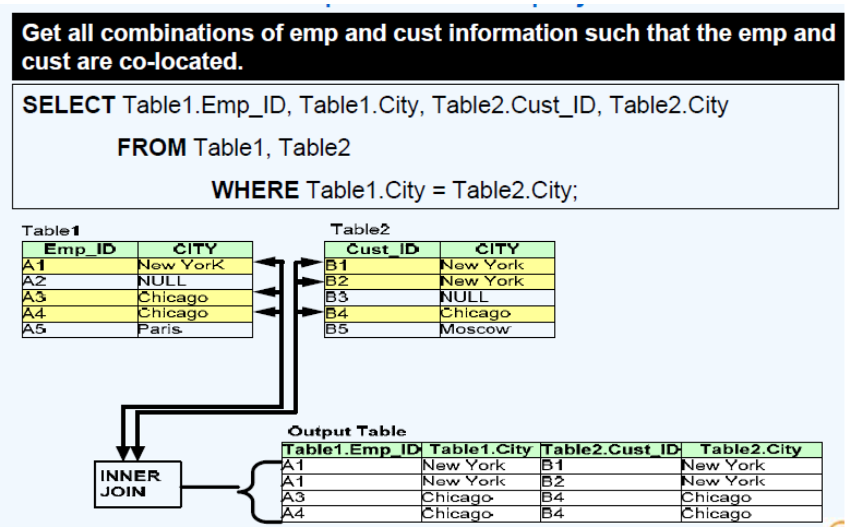
Cartesian product 14*4=56 rows selected

## Inner Join

An inner join between two (or more) tables is the Cartesian product that satisfies the join condition in the WHERE clause. It is of two types, depending upon the WHERE consition.

### Equality Joins

If the WHERE clause in query of relating two tables used an equality operator (=), it is an equality join, also known as an inner join or an equijoin. As illustrated below.



**Get all combinations of emp and cust information such that the emp and cust are co-located.**

```
SELECT Table1.Emp_ID, Table1.City, Table2.Cust_ID, Table2.City
        FROM Table1, Table2
                WHERE Table1.City = Table2.City;
```

Table1

| Emp_ID | CITY |
|---|---|
| A1 | New York |
| A2 | NULL |
| A3 | Chicago |
| A4 | Chicago |
| A5 | Paris |

Table2

| Cust_ID | CITY |
|---|---|
| B1 | New York |
| B2 | New York |
| B3 | NULL |
| B4 | Chicago |
| B5 | Moscow |

Output Table

| Table1.Emp_ID | Table1.City | Table2.Cust_ID | Table2.City |
|---|---|---|---|
| A1 | New York | B1 | New York |
| A1 | New York | B2 | New York |
| A3 | Chicago | B4 | Chicago |
| A4 | Chicago | B4 | Chicago |

The query discussed earlier to list the department name location for each employee, along with his or her salary is an example of equality join.

To get the required information following query is used:

```
SQL>SELECT dname, loc, ename, sal FROM dept, emp
            WHERE dept.deptno = emp.deptno;
```

### Non-Equality Join

If any other operator instead of equality operator (=) is used to join the tables in the query, it is non-equality join.

We have already seen examples of equality joins; let's consider an example of non-equality join.

**Example:**

**EMP**

| **EMPNO** | **ENAME** | **SAL** |
|---|---|---|
| 7839 | KING | 5000 |
| 7698 | BLAKE | 2850 |
| 7782 | CLARK | 2450 |
| 7566 | JONES | 2975 |
| 7654 | MARTIN | 1250 |
| 7499 | ALLEN | 1600 |
| 7844 | TURNER | 1500 |
| 7900 | JAMES | 950 |

↖

Salary in the EMP table is between low salary and high salary in the SALGRADE table

**SAL GRADE**

| **GRADE** | **LOSAL** | **HISAL** |
|---|---|---|
| 1 | 700 | 1200 |
| 2 | 1201 | 1400 |
| 3 | 1401 | 2000 |
| 4 | 2001 | 3000 |
| 5 | 3001 | 9999 |

The relationship between the EMP table and the SALGRADE table is a non-equijoin, meaning that no column in the EMP table corresponds directly to a column in the SALGRADE table. The relationship between the two tables is that the SAL column in the EMP table is between the LOSAL and HISAL column of the SALGRADE table. The relationship is obtained using an operator other than equal (=).

```
SQL> SELECT e.ename, e.sal, s.grade from emp e, salgrade s
          WHERE e.sal BETWEEN s.losal and S.hisal;
```

**OUTPUT:**

| **ENAME** | **SAL** | **GRADE** |
|---|---|---|
| JAMES | 950 | 1 |
| SMITH | 800 | 1 |
| ADAMS | 1100 | 1 |
| ……. | ……. | …… |

14 rows selected.

The above example creates a non-equijoin to evaluate an employee’s grade. The salary must be between any pair of the low and high salary ranges.