select Department.Deptname, Students.sname

from Department inner join Students

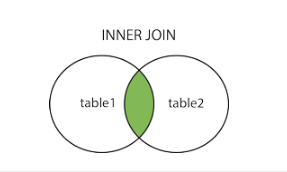
ON Department.Deptid= Students.Sid

The INNER JOIN keyword selects all rows from both tables as long as there is a match between the columns. If there are rows in the "Customers" table that do not have matches in "Orders", these customers will NOT be listed.

Different SQL JOINs

Before we continue with examples, we will list the types of the different SQL JOINs you can use:

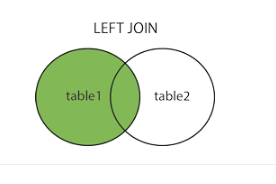
* **INNER JOIN**: Returns all rows when there is at least one match in BOTH tables
* **LEFT JOIN**: Return all rows from the left table, and the matched rows from the right table
* **RIGHT JOIN**: Return all rows from the right table, and the matched rows from the left table
* **FULL JOIN**: Return all rows when there is a match in ONE of the tables



SELECT Customers.CustomerName, Orders.OrderID  
FROM Customers  
INNER JOIN Orders  
ON Customers.CustomerID=Orders.CustomerID  
ORDER BY Customers.CustomerName;

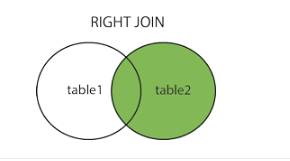
## SQL LEFT JOIN Keyword

The LEFT JOIN keyword returns all rows from the left table (table1), with the matching rows in the right table (table2). The result is NULL in the right side when there is no match.



SELECT Customers.CustomerName, Orders.OrderID  
FROM Customers  
LEFT JOIN Orders  
ON Customers.CustomerID=Orders.CustomerID  
ORDER BY Customers.CustomerName;

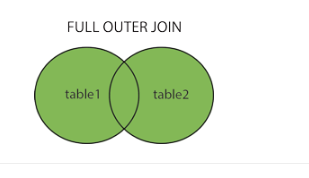
## SQL RIGHT JOIN Keyword



SELECT Orders.OrderID, Employees.FirstName  
FROM Orders  
RIGHT JOIN Employees  
ON Orders.EmployeeID=Employees.EmployeeID  
ORDER BY Orders.OrderID;

Employee will be viewed and intersection

## SQL FULL OUTER JOIN Keyword



SELECT Customers.CustomerName, Orders.OrderID  
FROM Customers  
FULL OUTER JOIN Orders  
ON Customers.CustomerID=Orders.CustomerID  
ORDER BY Customers.CustomerName;

## What is Self Join in SQL?

A self join is a join in which a table is joined with itself (which is also called Unary relationships), especially when the table has a FOREIGN KEY which references its own PRIMARY KEY. To join a table itself means that each row of the table is combined with itself and with every other row of the table.

The self join can be viewed as a join of two copies of the same table. The table is not actually copied, but SQL performs the command as though it were.

The syntax of the command for joining a table to itself is almost same as that for joining two different tables. To distinguish the column names from one another, aliases for the actual the table name are used, since both the tables have the same name. Table name aliases are defined in the FROM clause of the SELECT statement. See the syntax :

SELECT a.column\_name, b.column\_name...

FROM table1 a, table1 b

WHERE a.common\_filed = b.common\_field;

For this tutorial we have used a table EMPLOYEE, that has [one-to-many](http://www.w3resource.com/sql/question-answer.php) relationship

1. **SELECT** a.emp\_id **AS** "Emp\_ID",a.emp\_name **AS** "Employee Name",
2. b.emp\_id **AS** "Supervisor ID",b.emp\_name **AS** "Supervisor Name"
3. **FROM** employee a, employee b
4. **WHERE** a.emp\_supv = b.emp\_id;

### UNION Syntax

The UNION operator is used to combine the result-set of two or more SELECT statements.

* Each SELECT statement within UNION must have the same number of columns
* The columns must also have similar data types
* The columns in each SELECT statement must also be in the same order

### UNION Syntax

SELECT column\_name(s) FROM table1  
UNION  
SELECT column\_name(s) FROM table2;

### UNION ALL Syntax

The UNION operator selects only distinct values by default. To allow duplicate values, use UNION ALL:

SELECT column\_name(s) FROM table1  
UNION ALL  
SELECT column\_name(s) FROM table2;

Example

The following SQL statement returns the cities (only distinct values) from both the "Customers" and the "Suppliers" table:

SELECT City FROM Customers  
UNION  
SELECT City FROM Suppliers  
ORDER BY City;

The following SQL statement returns the cities (duplicate values also) from both the "Customers" and the "Suppliers" table:

SELECT City FROM Customers  
UNION ALL  
SELECT City FROM Suppliers  
ORDER BY City;

## SQL UNION With WHERE

The following SQL statement returns the German cities (only distinct values) from both the "Customers" and the "Suppliers" table:

SELECT City, Country FROM Customers  
WHERE Country='Germany'  
UNION  
SELECT City, Country FROM Suppliers  
WHERE Country='Germany'  
ORDER BY City;