==========

Module : 10 JOINS

==========

JOINS Types

Joins in SQL server are used to query (retrieve) data from 2 or more related tables.
In general tables are related to each other using foreign key constraints.

In SQL server, there are different types of JOINS.
1. CROSS JOIN
2. INNER JOIN
3. OUTER JOIN

Outer Joins are again divided into 3 types
1. Left Join or Left Outer Join
2. Right Join or Right Outer Join
3. Full Join or Full Outer Join

```
Create Database Emp_DB
Use Emp_DB;

Create table tblDepartment
(
     ID int primary key,
     DepartmentName nvarchar(50),
     Location nvarchar(50),
     DepartmentHead nvarchar(50)
);

Insert into tblDepartment values (1, 'IT', 'London', 'Rick')
Insert into tblDepartment values (2, 'Payroll', 'Delhi', 'Ron')
Insert into tblDepartment values (3, 'HR', 'New York', 'Christie')
Insert into tblDepartment values (4, 'Other Department', 'Sydney', 'Cindrella')


Create table tblEmployee
(
     ID int primary key,
     Name nvarchar(50),
     Gender nvarchar(50),
     Salary int,
     DepartmentId int foreign key references tblDepartment(Id)
);

Insert into tblEmployee values (1, 'Tom', 'Male', 4000, 1)
Insert into tblEmployee values (2, 'Pam', 'Female', 3000, 3)
Insert into tblEmployee values (3, 'John', 'Male', 3500, 1)
Insert into tblEmployee values (4, 'Sam', 'Male', 4500, 2)
Insert into tblEmployee values (5, 'Todd', 'Male', 2800, 2)
```

```
Insert into tblEmployee values (6, 'Ben', 'Male', 7000, 1)
Insert into tblEmployee values (7, 'Sara', 'Female', 4800, 3)
Insert into tblEmployee values (8, 'Valarie', 'Female', 5500, 1)
Insert into tblEmployee values (9, 'James', 'Male', 6500, NULL)
Insert into tblEmployee values (10, 'Russell', 'Male', 8800, NULL)

Select * from tblDepartment;
Select * from tblEmployee;

General Formula for Joins
---------------------------------
SELECT ColumnList
FROM LeftTableName
JOIN_TYPE RightTableName
ON JoinCondition

---------------------------------
1. CROSS JOIN:
---------------------------------
CROSS JOIN, produces the cartesian product of the 2 tables involved in the join.

CROSS JOIN Query:
---------------------------------
SELECT Name, Gender, Salary, DepartmentName
FROM tblEmployee
CROSS JOIN tblDepartment

SELECT *
FROM tblEmployee
CROSS JOIN tblDepartment

SELECT Name, Gender, Salary, DepartmentName
FROM tblEmployee, tblDepartment

---------------------------------
2. JOIN or INNER JOIN
---------------------------------
JOIN or INNER JOIN means the same.
It's always better to use INNER JOIN, as this explicitly specifies your intention.

Select * from tblDepartment;
Select * from tblEmployee;

INNER JOIN Query:
-------------------------------------
Write a query, to retrieve Name, Gender, Salary and DepartmentName from Employees
and Departments table.
SELECT Name, Gender, Salary, DepartmentName
FROM tblEmployee
INNER JOIN tblDepartment
ON tblEmployee.DepartmentId = tblDepartment.Id

OR
```

```
SELECT Name, Gender, Salary, DepartmentName
FROM tblEmployee
JOIN tblDepartment
ON tblEmployee.DepartmentId = tblDepartment.Id
```

So, in summary, INNER JOIN, returns only the matching rows between both the tables. Non matching rows are eliminated.

---

3. LEFT JOIN or LEFT OUTER JOIN

LEFT JOIN, returns all the matching rows + non matching rows from the left table. In reality, INNER JOIN and LEFT JOIN are extensively used.

LEFT JOIN Query:

Now, let's say, we want all the rows from the Employees table, including JAMES and RUSSELL records.

```
SELECT Name, Gender, Salary, DepartmentName
FROM tblEmployee
LEFT OUTER JOIN tblDepartment
ON tblEmployee.DepartmentId = tblDepartment.Id

OR

SELECT Name, Gender, Salary, DepartmentName
FROM tblEmployee
LEFT JOIN tblDepartment
ON tblEmployee.DepartmentId = tblDepartment.Id
```

Note: You can use, LEFT JOIN or LEFT OUTER JOIN. OUTER keyowrd is optional

---

4. RIGHT JOIN or RIGHT OUTER JOIN

RIGHT JOIN, returns all the matching rows + non matching rows from the right table.

RIGHT JOIN Query :

Let's say, we want, all the rows from the right table.

```
SELECT Name, Gender, Salary, DepartmentName
FROM tblEmployee
RIGHT OUTER JOIN tblDepartment
ON tblEmployee.DepartmentId = tblDepartment.Id;

OR

SELECT Name, Gender, Salary, DepartmentName
FROM tblEmployee
RIGHT JOIN tblDepartment
```

```
ON tblEmployee.DepartmentId = tblDepartment.Id
```

---

5. FULL JOIN or FULL OUTER JOIN

FULL JOIN, returns all rows from both the left and right tables, including the non matching rows.

FULL JOIN Query :

Let's say, want all the rows from both the tables involved in the join.

```
SELECT Name, Gender, Salary, DepartmentName
FROM tblEmployee
FULL OUTER JOIN tblDepartment
ON tblEmployee.DepartmentId = tblDepartment.Id

OR

SELECT Name, Gender, Salary, DepartmentName
FROM tblEmployee
FULL JOIN tblDepartment
ON tblEmployee.DepartmentId = tblDepartment.Id
```

---

5. Advanced or intelligent joins :

---

1. How to retrieve only the non matching rows from the left table.

```
SELECT Name, Gender, Salary, DepartmentName
FROM tblEmployee E
LEFT JOIN tblDepartment D
ON E.DepartmentId = D.Id
WHERE D.Id IS NULL;
```

2. How to retrieve only the non matching rows from the right table

```
SELECT Name, Gender, Salary, DepartmentName
FROM tblEmployee E
RIGHT JOIN tblDepartment D
ON E.DepartmentId = D.Id
WHERE E.DepartmentId IS NULL
```

3. How to retrieve only the non matching rows from both the left and right table.

```
SELECT Name, Gender, Salary, DepartmentName
FROM tblEmployee E
FULL JOIN tblDepartment D
ON E.DepartmentId = D.Id
WHERE E.DepartmentId IS NULL
OR D.Id IS NULL
```

---

6. SELF JOIN :

SELF JOIN is not a different type of JOIN, joining a table with itself is called as SELF JOIN
It can be classified under any type of JOIN - INNER, OUTER or CROSS Joins.

```
Create table tblEmployee2
(
     EmployeeId int primary key,
     Name nvarchar(50),
     ManagerId int,
);

Insert into tblEmployee2 values (1, 'Mike', 3)
Insert into tblEmployee2 values (2, 'Rob', 1)
Insert into tblEmployee2 values (3, 'Todd', Null)
Insert into tblEmployee2 values (4, 'Ben', 1)
Insert into tblEmployee2 values (5, 'Sam', 1)

Select * from tblEmployee2;
```

Left Self Join :

```
Select E.Name as Employee, M.Name as Manager_
from tblEmployee2 E_
Left Join tblEmployee2 M_
On E.ManagerId = M.EmployeeId
```

Inner Self Join :_

```
Select E.Name as Employee, M.Name as Manager_
from tblEmployee2 E_
Inner Join tblEmployee2 M_
On E.ManagerId = M.EmployeeId
```

Cross Self Join :_

```
Select E.Name as Employee, M.Name as Manager_
from tblEmployee2_E
Cross Join tblEmployee2 M
```