(T5)比較 Join 和 UNION。比較 ISNULL、CaseWhen、COALESCE
CourseGUID: e48417fc-9db5-4e99-822c-706c5ccef6cc
==========
(T5)比較 Join 和 UNION。比較 ISNULL、CaseWhen、COALESCE
==========

==========

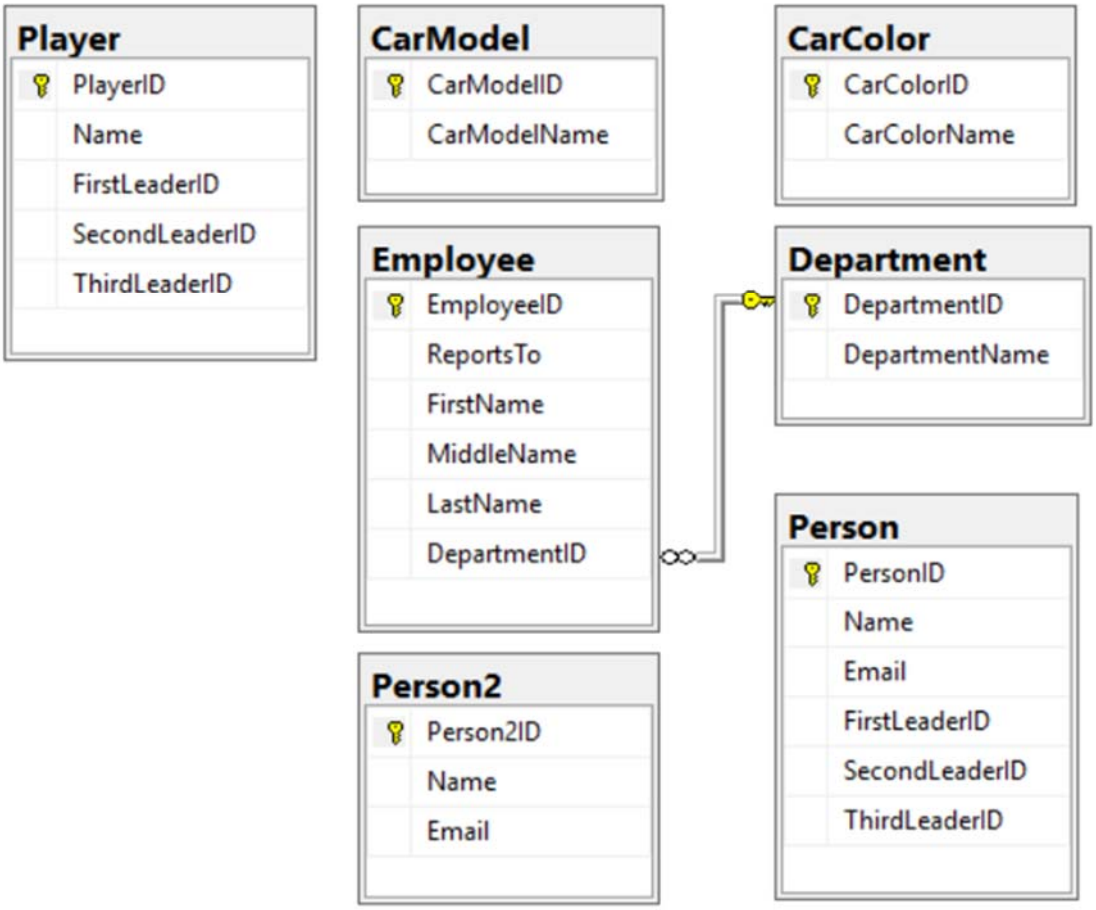
0. In Summary

In Summary
1. JOIN
1.1.
- (INNER) JOIN
Returns only the matching rows.
Non matching rows are eliminated.
1.2.
- LEFT (OUTER) JOIN
- LEFT (OUTER) JOIN - (INNER) JOIN
Returns all the matching rows +
non matching rows from the left table
1.3.
- RIGHT (OUTER) JOIN
- RIGHT (OUTER) JOIN - (INNER) JOIN
Returns all the matching rows +
non matching rows from the right table
1.4.
- FULL (OUTER) JOIN
- FULL (OUTER) JOIN - (INNER) JOIN
Returns all rows from both tables,
including the non-matching rows.
1.5.
- CROSS JOIN
Returns Cartesian product of the tables
involved in the join
CROSS JOIN does not need ON
1.6.
- Best SelfJoin_LEFT/Right (Outer) Join
- 2nd-Best SelfJoin_(INNER) JOIN
- Worst SelfJoin_CROSS Join - No sense

2.
2.1.
-ISNULL(A,B)

if A is NULL then B, if A is not NULL then A.
E.g.
--SELECT ISNULL(NULL, 'No Manager') AS ManagerFullName;
--SELECT ISNULL('Name1', 'No Manager') AS ManagerFullName;
--SELECT COALESCE(NULL, 'No Manager') AS ManagerFullName;
--SELECT COALESCE('Name1', 'No Manager') AS ManagerFullName;
2.2.
-CASE WHEN Expression THEN 'A' ELSE 'B' END
if expression is true, then A otherwise B
E.g.
--CASE WHEN (m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName) IS NULL
--   THEN 'No Manager'
--   ELSE (m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName)
--END
2.3.
-COALESCE(A, B, C ...etc)
Return the first non-NULL value.
The way to remember, COALESCE, is "Coal Ese E".
E.g.
--SELECT COALESCE(NULL, 'A', 'B') AS FirstAvailableLeader;
Return A
--SELECT COALESCE(NULL, NULL, 'B', 'A') AS FirstAvailableLeader;
Return B
--SELECT COALESCE('D', NULL, 'B', 'C') AS FirstAvailableLeader;
Return D

3.
UNION V.S. UNION ALL
3.1.
UNION removes duplicate rows,
UNION ALL does not.
3.2.
ORDER BY caluse can only be used on the last SELECT statement.
ORDER BY caluse is on any other SELECT statement will cause Syntax Error.
UNION combines rows from 2 or more tables/Search Results.
Thus, ORDER BY caluse can only be used after all the results is combined.
3.3.
UNION combines rows from 2 or more tables/Search Results.
JOINS combine columns from 2 or more tables.



==========

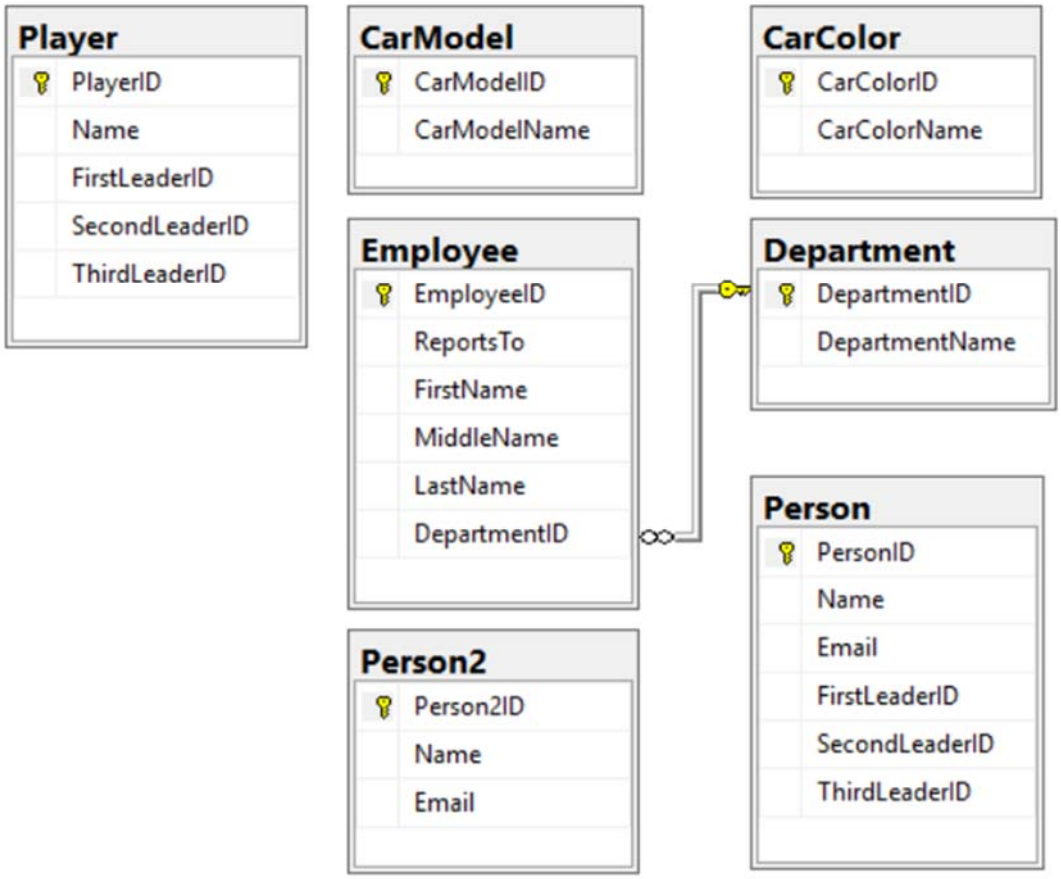
1. Query - Cross/Inner/(Left/Right/Full) Outter/Self Join Joins

What to learn in this part

1. JOIN
1.1.
- (INNER) JOIN
Returns only the matching rows.
Non matching rows are eliminated.
1.2.
- LEFT (OUTER) JOIN
- LEFT (OUTER) JOIN - (INNER) JOIN
Returns all the matching rows +
non matching rows from the left table
1.3.
- RIGHT (OUTER) JOIN
- RIGHT (OUTER) JOIN - (INNER) JOIN
Returns all the matching rows +
non matching rows from the right table
1.4.
- FULL (OUTER) JOIN

- FULL (OUTER) JOIN - (INNER) JOIN
Returns all rows from both tables,
including the non-matching rows.
1.5.
- CROSS JOIN
Returns Cartesian product of the tables
involved in the join
CROSS JOIN does not need ON
1.6.
- Best SelfJoin_LEFT/Right (Outer) Join
- 2nd-Best SelfJoin_(INNER) JOIN
- Worst SelfJoin_CROSS Join - No sense

The Sample Database diagram as the following



The sample database script as the following.

```
--=============================================================================
--T005_01_(INNER)JOIN
--=============================================================================
--=============================================================================
--T005_01_01
--Create Sample Data
--If Table exists then DROP it
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'Employee' ) )
   BEGIN
       TRUNCATE TABLE Employee;
       DROP TABLE Employee;
```

```
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'Department' ) )
    BEGIN
        TRUNCATE TABLE Department;
        DROP TABLE Department;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'CarModel' ) )
    BEGIN
        TRUNCATE TABLE CarModel;
        DROP TABLE CarModel;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'CarColor' ) )
    BEGIN
        TRUNCATE TABLE CarColor;
        DROP TABLE CarColor;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'Player' ) )
    BEGIN
        TRUNCATE TABLE Player;
        DROP TABLE Player;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'Person' ) )
    BEGIN
        TRUNCATE TABLE Person;
        DROP TABLE Person;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'Person2' ) )
    BEGIN
        TRUNCATE TABLE Person2;
        DROP TABLE Person2;
    END;
GO -- Run the previous command and begins new batch
CREATE TABLE Department
(
  DepartmentID INT IDENTITY(1, 1)
```

```
                    PRIMARY KEY
                    NOT NULL ,
  DepartmentName NVARCHAR(50) NULL,
);
GO -- Run the previous command and begins new batch
INSERT  [dbo].[Department]
VALUES  ( N'Department1' );
INSERT  [dbo].[Department]
VALUES  ( N'Department2' );
INSERT  [dbo].[Department]
VALUES  ( N'Department3' );
INSERT  [dbo].[Department]
VALUES  ( N'Department4' );
INSERT  [dbo].[Department]
VALUES  ( N'Department5' );
INSERT  [dbo].[Department]
VALUES  ( N'Department6' );
GO -- Run the previous command and begins new batch
CREATE TABLE Employee
(
  EmployeeID INT IDENTITY(1, 1)
                  PRIMARY KEY
                  NOT NULL ,
  ReportsTo INT NULL ,
  FirstName NVARCHAR(100) NULL ,
  MiddleName NVARCHAR(100) NULL ,
  LastName NVARCHAR(100) NULL ,
  DepartmentID INT FOREIGN KEY REFERENCES Department ( DepartmentID )
                  NULL
);
GO -- Run the previous command and begins new batch
INSERT  Employee
VALUES  ( NULL, N'First1', N'Middle1', N'Last1', 1 );
INSERT  Employee
VALUES  ( 1, N'First2', N'Middle2', N'Last2', 2 );
INSERT  Employee
VALUES  ( 1, N'Fisrt3', N'Middle3', N'Last3', 3 );
INSERT  Employee
VALUES  ( 2, N'First4', N'Middle4', N'Last4', 1 );
INSERT  Employee
VALUES  ( 2, N'First5', N'Middle5', N'Last5', 2 );
INSERT  Employee
VALUES  ( 2, N'First6', N'Middle6', N'Last6', 3 );
INSERT  Employee
VALUES  ( 3, N'First7', N'Middle7', N'Last7', 1 );
INSERT  Employee
VALUES  ( 3, N'First8', N'Middle8', N'Last8', 2 );
INSERT  Employee
VALUES  ( 3, N'First9', N'Middle9', N'last9', NULL );
INSERT  Employee
VALUES  ( NULL, N'First10', N'Middle10', N'Last10', NULL );
GO -- Run the previous command and begins new batch
CREATE TABLE CarColor
(
  CarColorID INT IDENTITY(1, 1)
                  PRIMARY KEY
                  NOT NULL ,
```

```
  CarColorName NVARCHAR(100) NULL,
);
GO -- Run the previous command and begins new batch
INSERT  CarColor
VALUES  ( N'Green' );
INSERT  CarColor
VALUES  ( N'Blue' );
INSERT  CarColor
VALUES  ( N'Red' );
GO -- Run the previous command and begins new batch
CREATE TABLE CarModel
(
  CarModelID INT IDENTITY(1, 1)
                 PRIMARY KEY
                 NOT NULL ,
  CarModelName NVARCHAR(100) NULL,
);
GO -- Run the previous command and begins new batch
INSERT  CarModel
VALUES  ( N'Toyota Yaris' );
INSERT  CarModel
VALUES  ( N'Toyota Corolla' );
INSERT  CarModel
VALUES  ( N'Toyota Camry' );
GO -- Run the previous command and begins new batch
CREATE TABLE Player
(
  PlayerID INT IDENTITY(1, 1)
               PRIMARY KEY
               NOT NULL ,
  [Name] NVARCHAR(100) NULL ,
  FirstLeaderID INT NULL ,
  SecondLeaderID INT NULL ,
  ThirdLeaderID INT NULL,
);
GO -- Run the previous command and begins new batch
INSERT  Player
VALUES  ( N'Name1', NULL, NULL, NULL );
INSERT  Player
VALUES  ( N'Name2', NULL, 1, NULL );
INSERT  Player
VALUES  ( N'Name3', NULL, 1, 2 );
INSERT  Player
VALUES  ( N'Name4', 1, 2, 3 );
INSERT  Player
VALUES  ( N'Name5', NULL, NULL, 1 );
INSERT  Player
VALUES  ( N'Name6', NULL, 2, 3 );
INSERT  Player
VALUES  ( N'Name7', NULL, NULL, 3 );
INSERT  Player
VALUES  ( N'Name8', NULL, 1, 2 );
INSERT  Player
VALUES  ( N'Name9', 1, 2, 3 );
INSERT  Player
VALUES  ( N'Name10', NULL, 1, 2 );
GO -- Run the previous command and begins new batch
```

```
CREATE TABLE Person
(
  PersonID INT IDENTITY(1, 1)
               PRIMARY KEY
               NOT NULL ,
  [Name] NVARCHAR(100) NULL ,
  Email NVARCHAR(500) NULL ,
  FirstLeaderID INT NULL ,
  SecondLeaderID INT NULL ,
  ThirdLeaderID INT NULL,
);
GO -- Run the previous command and begins new batch
INSERT  Person
VALUES  ( N'Name1', N'1@1.com', NULL, NULL, NULL );
INSERT  Person
VALUES  ( N'Name2', N'2@2.com', NULL, 1, NULL );
INSERT  Person
VALUES  ( N'Name3', N'3@3.com', NULL, 1, 2 );
INSERT  Person
VALUES  ( N'Name4', N'4@4.com', 1, 2, 3 );
INSERT  Person
VALUES  ( N'Name5', N'5@5.com', NULL, NULL, 1 );
INSERT  Person
VALUES  ( N'Name6', N'6@6.com', NULL, 2, 3 );
INSERT  Person
VALUES  ( N'Name7', N'7@7.com', NULL, NULL, 3 );
INSERT  Person
VALUES  ( N'Name8', N'8@8.com', NULL, 1, 2 );
INSERT  Person
VALUES  ( N'Name9', N'9@9.com', 1, 2, 3 );
INSERT  Person
VALUES  ( N'Name10', N'10@10.com', NULL, 1, 2 );
GO -- Run the previous command and begins new batch
CREATE TABLE Person2
(
  Person2ID INT IDENTITY(1, 1)
                PRIMARY KEY
                NOT NULL ,
  [Name] NVARCHAR(100) NULL ,
  Email NVARCHAR(500) NULL,
);
GO -- Run the previous command and begins new batch
INSERT  Person2
VALUES  ( N'Name6', N'6@6.com' );
INSERT  Person2
VALUES  ( N'Name7', N'7@7.com' );
INSERT  Person2
VALUES  ( N'Name8', N'8@8.com' );
INSERT  Person2
VALUES  ( N'Name9', N'9@9.com' );
INSERT  Person2
VALUES  ( N'Name10', N'10@10.com' );
INSERT  Person2
VALUES  ( N'Name11', N'11@11.com' );
INSERT  Person2
VALUES  ( N'Name12', N'12@12.com' );
INSERT  Person2
```

```
VALUES  ( N'Name13', N'13@13.com' );
INSERT  Person2
VALUES  ( N'Name14', N'14@14.com' );
INSERT  Person2
VALUES  ( N'Name15', N'15@15.com' );
GO -- Run the previous command and begins new batch
SELECT  *
FROM    Department;
```

	DepartmentID	DepartmentName
1	1	Department1
2	2	Department2
3	3	Department3
4	4	Department4
5	5	Department5
6	6	Department6

```
SELECT  *
FROM    Employee;
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

```
SELECT  *
FROM    CarColor;
```

	CarColorID	CarColorName
1	1	Green
2	2	Blue
3	3	Red

```
SELECT  *
FROM    CarModel;
```

	CarModelID	CarModelName
1	1	Toyota Yaris
2	2	Toyota Corolla
3	3	Toyota Camry

```
SELECT  *
FROM    Player;
```

	PlayerID	Name	FirstLeaderID	SecondLeaderID	ThirdLeaderID
1	1	Name1	NULL	NULL	NULL
2	2	Name2	NULL	1	NULL
3	3	Name3	NULL	1	2
4	4	Name4	1	2	3
5	5	Name5	NULL	NULL	1
6	6	Name6	NULL	2	3
7	7	Name7	NULL	NULL	3
8	8	Name8	NULL	1	2
9	9	Name9	1	2	3
10	10	Name10	NULL	1	2

```
SELECT  *
FROM    Person;
```

	PersonID	Name	Email	FirstLeaderID	SecondLeaderID	ThirdLeaderID
1	1	Name1	1@1.com	NULL	NULL	NULL
2	2	Name2	2@2.com	NULL	1	NULL
3	3	Name3	3@3.com	NULL	1	2
4	4	Name4	4@4.com	1	2	3
5	5	Name5	5@5.com	NULL	NULL	1
6	6	Name6	6@6.com	NULL	2	3
7	7	Name7	7@7.com	NULL	NULL	3
8	8	Name8	8@8.com	NULL	1	2
9	9	Name9	9@9.com	1	2	3
10	10	Name10	10@10.com	NULL	1	2

```
SELECT  *
FROM    Person2;
GO -- Run the previous command and begins new batch
```

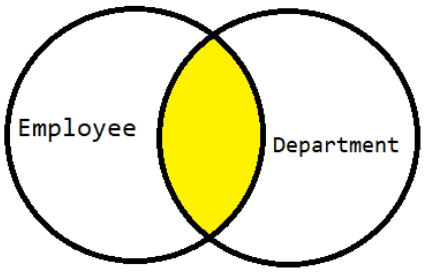
	Person2ID	Name	Email
1	1	Name6	6@6.com
2	2	Name7	7@7.com
3	3	Name8	8@8.com
4	4	Name9	9@9.com
5	5	Name10	10@10.com
6	6	Name11	11@11.com
7	7	Name12	12@12.com
8	8	Name13	13@13.com
9	9	Name14	14@14.com
10	10	Name15	15@15.com

==========

1.1. (INNER)JOIN

```
--=============================================================================
--T005_01_02
--(INNER) JOIN
SELECT  *
FROM    Employee;
SELECT  *
FROM    Department;
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        d.DepartmentName
FROM    dbo.Employee e
        INNER JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
        -- JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
GO -- Run the prvious command and begins new batch
/*
1.
(INNER) JOIN
Returns only the matching rows.
Non matching rows are eliminated.
2.
Employee has 10 rows
Department has 6 rows
There are 8 matching rows.
There are 2 non-matching rows from Employee.
```

```
There are 3 non-matching rows from Department.
-->
(INNER) JOIN will only show 8 matching rows.
*/
```



(INNER) JOIN

```
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        d.DepartmentName
FROM    dbo.Employee e
        INNER JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
        -- JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;

/*
1.
(INNER) JOIN
Returns only the matching rows.
Non matching rows are eliminated.
2.
Employee has 10 rows
Department has 6 rows
There are 8 matching rows.
There are 2 non-matching rows from Employee.
There are 3 non-matching rows from Department.
-->
(INNER) JOIN will only show 8 matching rows.
*/
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

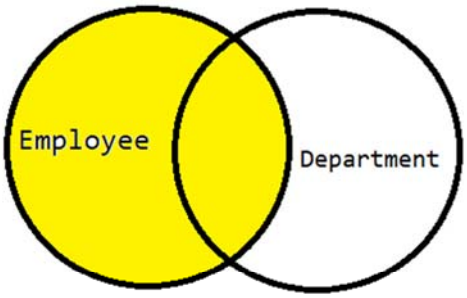
	DepartmentID	DepartmentName
1	1	Department1
2	2	Department2
3	3	Department3
4	4	Department4
5	5	Department5
6	6	Department6

	FullName	DepartmentName
1	First1 Middle1 Last1	Department1
2	First2 Middle2 Last2	Department2
3	Fisrt3 Middle3 Last3	Department3
4	First4 Middle4 Last4	Department1
5	First5 Middle5 Last5	Department2
6	First6 Middle6 Last6	Department3
7	First7 Middle7 Last7	Department1
8	First8 Middle8 Last8	Department2

==========

1.2. LEFT (OUTER) JOIN

```
--============================================================================
--T005_02_LEFT (OUTER) JOIN
--============================================================================
--============================================================================
--T005_02_01
--LEFT (OUTER) JOIN
SELECT  *
FROM    Employee;
SELECT  *
FROM    Department;
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        d.DepartmentName
FROM    dbo.Employee e
        LEFT OUTER JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
        --LEFT JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
GO -- Run the prvious command and begins new batch
/*
1.
LEFT (OUTER) JOIN
Returns all the matching rows +
non matching rows from the left table
2.
Employee has 10 rows
Department has 6 rows
There are 8 matching rows.
There are 2 non-matching rows from Employee.
There are 3 non-matching rows from Department.
-->
LEFT (OUTER) JOIN will show
(8 matching rows + 2 non-matching rows from Employee).
*/
```



LEFT (OUTER) JOIN

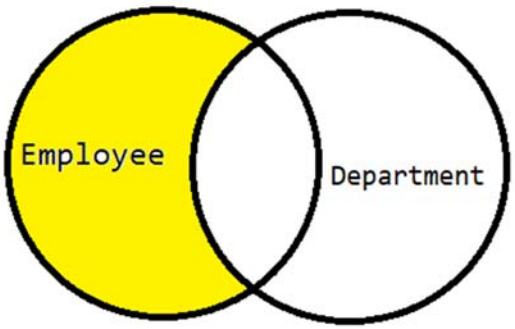
```
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        d.DepartmentName
FROM    dbo.Employee e
        LEFT OUTER JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
        --LEFT JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;

/*
1.
LEFT (OUTER) JOIN
Returns all the matching rows +
non matching rows from the left table
2.
Employee has 10 rows
Department has 6 rows
There are 8 matching rows.
There are 2 non-matching rows from Employee.
There are 3 non-matching rows from Department.
-->
LEFT (OUTER) JOIN will show
(8 matching rows + 2 non-matching rows from Employee).
*/
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

	DepartmentID	DepartmentName
1	1	Department1
2	2	Department2
3	3	Department3
4	4	Department4
5	5	Department5
6	6	Department6

	FullName	DepartmentName
1	First1 Middle1 Last1	Department1
2	First2 Middle2 Last2	Department2
3	Fisrt3 Middle3 Last3	Department3
4	First4 Middle4 Last4	Department1
5	First5 Middle5 Last5	Department2
6	First6 Middle6 Last6	Department3
7	First7 Middle7 Last7	Department1
8	First8 Middle8 Last8	Department2
9	First9 Middle9 last9	NULL
10	First10 Middle10 Last10	NULL

```
--=============================================================================
--T005_02_02
--LEFT (OUTER) JOIN - (INNER) JOIN
USE Sample;
GO -- Run the prvious command and begins new batch
SELECT  *
FROM    Employee;
SELECT  *
FROM    Department;
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        d.DepartmentName
FROM    dbo.Employee e
        LEFT OUTER JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID
             --LEFT JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
WHERE   d.DepartmentID IS NULL;
GO -- Run the prvious command and begins new batch
/*
1.
LEFT (OUTER) JOIN
Returns all the matching rows +
non matching rows from the left table
2.
Employee has 10 rows
Department has 6 rows
There are 8 matching rows.
There are 2 non-matching rows from Employee.
There are 3 non-matching rows from Department.
-->
LEFT (OUTER) JOIN will show
(8 matching rows + 2 non-matching rows from Employee).
3.
--WHERE   d.DepartmentID IS NULL;
This is eliminate (8 matching rows).
Thus, only show   (2 non-matching rows from Employee)
*/
```



Left (Outer) Join
- (Inner) Join

```
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        d.DepartmentName
FROM    dbo.Employee e
        LEFT OUTER JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID
              --LEFT JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
WHERE   d.DepartmentID IS NULL;
```

```
/*
1.
LEFT (OUTER) JOIN
Returns all the matching rows +
non matching rows from the left table
2.
Employee has 10 rows
Department has 6 rows
There are 8 matching rows.
There are 2 non-matching rows from Employee.
There are 3 non-matching rows from Department.
-->
LEFT (OUTER) JOIN will show
(8 matching rows + 2 non-matching rows from Employee).
3.
--WHERE   d.DepartmentID IS NULL;
This is eliminate (8 matching rows).
Thus, only show   (2 non-matching rows from Employee)
*/
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

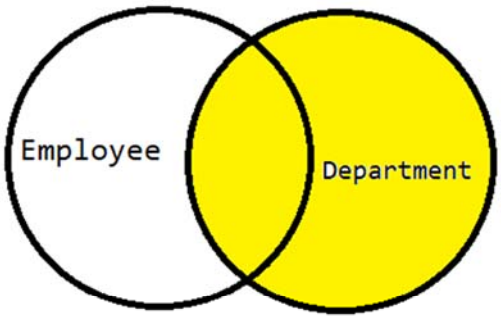
	DepartmentID	DepartmentName
1	1	Department1
2	2	Department2
3	3	Department3
4	4	Department4
5	5	Department5
6	6	Department6

	FullName	DepartmentName
1	First9 Middle9 last9	NULL
2	First10 Middle10 Last10	NULL

==========

1.3. RIGHT (OUTER) JOIN

```
--============================================================================
--T005_03_RIGHT (OUTER) JOIN
--============================================================================
--============================================================================
--T005_03_01
--RIGHT (OUTER) JOIN
SELECT  *
FROM    Employee;
SELECT  *
FROM    Department;
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        d.DepartmentName
FROM    dbo.Employee e
        RIGHT OUTER JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
        --RIGHT JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
GO -- Run the prvious command and begins new batch
/*
1.
--RIGHT (OUTER) JOIN
Returns all the matching rows +
non matching rows from the right table
2.
Employee has 10 rows
Department has 6 rows
There are 8 matching rows.
There are 2 non-matching rows from Employee.
There are 3 non-matching rows from Department.
-->
LEFT (OUTER) JOIN will show
(8 matching rows + 3 non-matching rows from Department).
*/
```



RIGHT (OUTER) JOIN

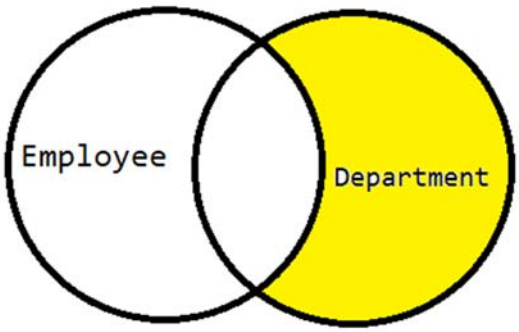
```
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        d.DepartmentName
FROM    dbo.Employee e
        RIGHT OUTER JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
        --RIGHT JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;

/*
1.
--RIGHT (OUTER) JOIN
Returns all the matching rows +
non matching rows from the right table
 2.
 Employee has 10 rows
 Department has 6 rows
 There are 8 matching rows.
 There are 2 non-matching rows from Employee.
 There are 3 non-matching rows from Department.
 -->
 LEFT (OUTER) JOIN will show
 (8 matching rows + 3 non-matching rows from
 Department).
 */
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

	DepartmentID	DepartmentName
1	1	Department1
2	2	Department2
3	3	Department3
4	4	Department4
5	5	Department5
6	6	Department6

	FullName	DepartmentName
1	First1 Middle1 Last1	Department1
2	First4 Middle4 Last4	Department1
3	First7 Middle7 Last7	Department1
4	First2 Middle2 Last2	Department2
5	First5 Middle5 Last5	Department2
6	First8 Middle8 Last8	Department2
7	Fisrt3 Middle3 Last3	Department3
8	First6 Middle6 Last6	Department3
9	NULL	Department4
10	NULL	Department5
11	NULL	Department6

```
--=============================================================================
--T005_03_02
--RIGHT (OUTER) JOIN - (INNER) JOIN
SELECT  *
FROM    Employee;
SELECT  *
FROM    Department;
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        d.DepartmentName
FROM    dbo.Employee e
        RIGHT OUTER JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID
              --RIGHT JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
WHERE   e.DepartmentID IS NULL;
GO -- Run the prvious command and begins new batch
--LEFT (OUTER) JOIN - INNER JOIN
/*
1.
--RIGHT (OUTER) JOIN
Returns all the matching rows +
non matching rows from the right table
2.
Employee has 10 rows
Department has 6 rows
There are 8 matching rows.
There are 2 non-matching rows from Employee.
There are 3 non-matching rows from Department.
-->
LEFT (OUTER) JOIN will show
(8 matching rows + 3 non-matching rows from Department).
3.
--WHERE   e.DepartmentID IS NULL;
This is eliminate (8 matching rows).
Thus, only show   (3 non-matching rows from Department)
*/
```



RIGHT (OUTER) JOIN
- (INNER) JOIN

```
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        d.DepartmentName
FROM    dbo.Employee e
        RIGHT OUTER JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID
              --RIGHT JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
WHERE   e.DepartmentID IS NULL;
/*
1.
--RIGHT (OUTER) JOIN
Returns all the matching rows +
non matching rows from the right table
2.
Employee has 10 rows
Department has 6 rows
There are 8 matching rows.
There are 2 non-matching rows from Employee.
There are 3 non-matching rows from Department.
-->
LEFT (OUTER) JOIN will show
(8 matching rows + 3 non-matching rows from Department).
3.
--WHERE   e.DepartmentID IS NULL;
This is eliminate (8 matching rows).
Thus, only show   (3 non-matching rows from Department)
*/
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

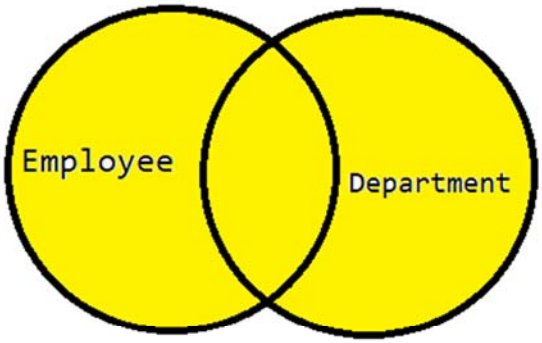
	DepartmentID	DepartmentName
1	1	Department1
2	2	Department2
3	3	Department3
4	4	Department4
5	5	Department5
6	6	Department6

	FullName	DepartmentName
1	NULL	Department4
2	NULL	Department5
3	NULL	Department6

==========

1.4. FULL (OUTER) JOIN

```
--============================================================================
--T005_04_FULL (OUTER) JOIN
--============================================================================
--============================================================================
--T005_04_01
--FULL (OUTER) JOIN
SELECT  *
FROM    Employee;
SELECT  *
FROM    Department;
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        d.DepartmentName
FROM    dbo.Employee e
        FULL OUTER JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
        --FULL JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
GO -- Run the prvious command and begins new batch
/*
1.
--FULL (OUTER) JOIN
Returns all rows from both tables,
including the non-matching rows.
2.
Employee has 10 rows
Department has 6 rows
There are 8 matching rows.
There are 2 non-matching rows from Employee.
There are 3 non-matching rows from Department.
-->
LEFT (OUTER) JOIN will show
(8 matching rows
+ 2 non-matching rows from Employee
+ 3 non-matching rows from Department).
*/
```



FULL (OUTER) JOIN

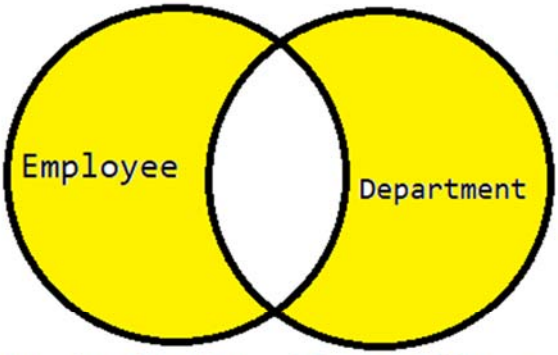
```
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        d.DepartmentName
FROM    dbo.Employee e
        FULL OUTER JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
        --FULL JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;

/*
1.
--FULL (OUTER) JOIN
Returns all rows from both tables,
including the non-matching rows.
2.
Employee has 10 rows
Department has 6 rows
There are 8 matching rows.
There are 2 non-matching rows from Employee.
There are 3 non-matching rows from Department.
-->
LEFT (OUTER) JOIN will show
(8 matching rows
+ 2 non-matching rows from Employee
+ 3 non-matching rows from Department).
*/
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

	DepartmentID	DepartmentName
1	1	Department1
2	2	Department2
3	3	Department3
4	4	Department4
5	5	Department5
6	6	Department6

	FullName	DepartmentName
1	First1 Middle1 Last1	Department1
2	First2 Middle2 Last2	Department2
3	Fisrt3 Middle3 Last3	Department3
4	First4 Middle4 Last4	Department1
5	First5 Middle5 Last5	Department2
6	First6 Middle6 Last6	Department3
7	First7 Middle7 Last7	Department1
8	First8 Middle8 Last8	Department2
9	First9 Middle9 last9	NULL
10	First10 Middle10 Last10	NULL
11	NULL	Department4
12	NULL	Department5
13	NULL	Department6

```
--================================================================================
--T005_04_02
--FULL (OUTER) JOIN - (INNER) JOIN
USE Sample;
GO -- Run the prvious command and begins new batch
SELECT  *
FROM    Employee;
SELECT  *
FROM    Department;
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        d.DepartmentName
FROM    dbo.Employee e
        FULL OUTER JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID
              --FULL JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
WHERE   e.DepartmentID IS NULL
        OR d.DepartmentID IS NULL;
GO -- Run the prvious command and begins new batch
/*
1.
--FULL (OUTER) JOIN
Returns all rows from both tables,
including the non-matching rows.
2.
Employee has 10 rows
Department has 6 rows
There are 8 matching rows.
There are 2 non-matching rows from Employee.
There are 3 non-matching rows from Department.
-->
LEFT (OUTER) JOIN will show
(8 matching rows
+ 2 non-matching rows from Employee
+ 3 non-matching rows from Department).
3.
--WHERE   e.DepartmentID IS NULL;
This is eliminate (8 matching rows).
Thus, only show
(2 non-matching rows from Employee
+ 3 non-matching rows from Department)
*/
```



FULL (OUTER) JOIN - (INNER) JOIN

```
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        d.DepartmentName
FROM    dbo.Employee e
        FULL OUTER JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID
              --FULL JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID;
WHERE   e.DepartmentID IS NULL
        OR d.DepartmentID IS NULL;

/*
1.
--FULL (OUTER) JOIN
Returns all rows from both tables,
including the non-matching rows.
2.
Employee has 10 rows
Department has 6 rows
There are 8 matching rows.
There are 2 non-matching rows from Employee.
There are 3 non-matching rows from Department.
-->
LEFT (OUTER) JOIN will show
(8 matching rows
+ 2 non-matching rows from Employee
+ 3 non-matching rows from Department).
3.
--WHERE   e.DepartmentID IS NULL;
This is eliminate (8 matching rows).
Thus, only show
(2 non-matching rows from Employee
+ 3 non-matching rows from Department)
*/
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

	DepartmentID	DepartmentName
1	1	Department1
2	2	Department2
3	3	Department3
4	4	Department4
5	5	Department5
6	6	Department6

	FullName	DepartmentName
1	First9 Middle9 last9	NULL
2	First10 Middle10 Last10	NULL
3	NULL	Department4
4	NULL	Department5
5	NULL	Department6

==========

1.5. CROSS JOIN

```
--=============================================================================
--T005_05_CROSS JOIN
--=============================================================================
--=============================================================================
--T005_05_01
--CROSS JOIN
SELECT  *
FROM    Employee;
SELECT  *
FROM    Department;
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        d.DepartmentName
FROM    dbo.Employee e
        CROSS JOIN dbo.Department d;
GO -- Run the prvious command and begins new batch
/*
1.
-- CROSS JOIN
Returns Cartesian product of the tables
involved in the join.
CROSS JOIN does not need ON
2.
Employee has 10 rows
Department has 6 rows
There are 8 matching rows.
There are 2 non-matching rows from Employee.
There are 3 non-matching rows from Department.
-->
CROSS JOIN will show
(10 rows from Employee) * (6 rows from Department) = 60 rows
*/
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

	DepartmentID	DepartmentName
1	1	Department1
2	2	Department2
3	3	Department3
4	4	Department4
5	5	Department5
6	6	Department6

	FullName	DepartmentName
1	First1 Middle1 Last1	Department1
2	First2 Middle2 Last2	Department1
3	Fisrt3 Middle3 Last3	Department1
4	First4 Middle4 Last4	Department1
5	First5 Middle5 Last5	Department1
6	First6 Middle6 Last6	Department1
7	First7 Middle7 Last7	Department1
8	First8 Middle8 Last8	Department1
9	First9 Middle9 last9	Department1
10	First10 Middle10 Last10	Department1
11	First1 Middle1 Last1	Department2
12	First2 Middle2 Last2	Department2
13	Fisrt3 Middle3 Last3	Department2
14	First4 Middle4 Last4	Department2
15	First5 Middle5 Last5	Department2
16	First6 Middle6 Last6	Department2
17	First7 Middle7 Last7	Department2
18	First8 Middle8 Last8	Department2
19	First9 Middle9 last9	Department2

N550JKL\lpmpl (52) | Sample | 00:00:00 | 60 rows

```
--=============================================================================
--T005_05_02
--CROSS JOIN
SELECT  *
FROM    CarColor;
SELECT  *
FROM    CarModel;
SELECT  ( cm.CarModelName + ' ' + cc.CarColorName ) AS CarList
FROM    dbo.CarColor cc
        CROSS JOIN dbo.CarModel cm;
GO -- Run the prvious command and begins new batch
/*
1.
-- CROSS JOIN
Returns Cartesian product of the tables
involved in the join.
CROSS JOIN does not need ON
```

```
2.
CarColor has 3 rows
CarModel has 3 rows
-->
CROSS JOIN will show
(3 rows from CarColor) * (3 rows from CarModel) = 9 rows
*/
```

CarColorID	CarColorName
1	Green
2	Blue
3	Red

	CarModelID	CarModelName
1	1	Toyota Yaris
2	2	Toyota Corolla
3	3	Toyota Camry

	CarList
1	Toyota Yaris Green
2	Toyota Yaris Blue
3	Toyota Yaris Red
4	Toyota Corolla Gr...
5	Toyota Corolla Blue
6	Toyota Corolla Red
7	Toyota Camry Gr...
8	Toyota Camry Blue
9	Toyota Camry Red

CROSS JOIN

```
SELECT  ( cm.CarModelName + ' ' + cc.CarColorName ) AS CarList
FROM    dbo.CarColor cc
        CROSS JOIN dbo.CarModel cm;

/*
1.
-- CROSS JOIN
Returns Cartesian product of the tables
involved in the join.
CROSS JOIN does not need ON
2.
CarColor has 3 rows
CarModel has 3 rows
-->
CROSS JOIN will show
(3 rows from CarColor) * (3 rows from CarModel) = 9 rows
*/
```

Results Messages

	CarColorID	CarColorName
1	1	Green
2	2	Blue
3	3	Red

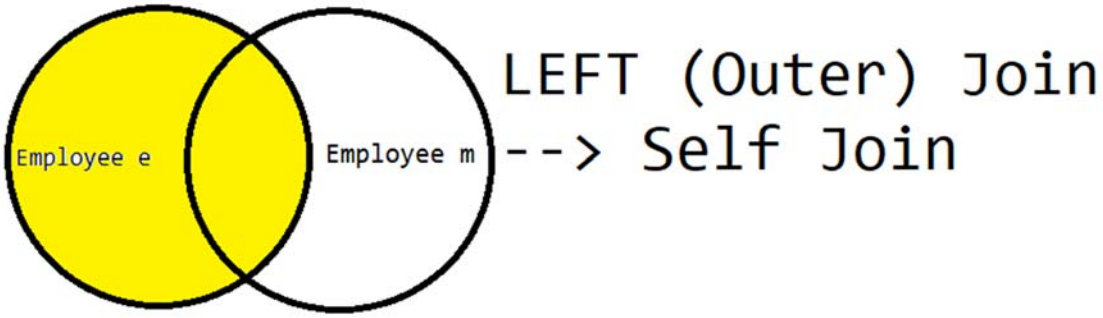
	CarModelID	CarModelName
1	1	Toyota Yaris
2	2	Toyota Corolla
3	3	Toyota Camry

	CarList
1	Toyota Yaris Green
2	Toyota Yaris Blue
3	Toyota Yaris Red
4	Toyota Corolla Gr...
5	Toyota Corolla Blue
6	Toyota Corolla Red
7	Toyota Camry Gr...
8	Toyota Camry Blue
9	Toyota Camry Red

1.6. SelfJoin_LEFT/Right (Outer) Join

```
--=============================================================================
--T005_06_SelfJoin_LEFT/Right (Outer) Join
--=============================================================================
--=============================================================================
--T005_06_01
--Best SelfJoin_LEFT/Right (Outer) Join
SELECT  *
FROM    Employee;
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) AS ManagerFullName
FROM    dbo.Employee e
        LEFT OUTER JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
             --LEFT JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
GO -- Run the prvious command and begins new batch
/*
1.
Best SelfJoin_LEFT/Right (Outer) Join
2.
- LEFT (OUTER) JOIN
Returns all the matching rows +
non matching rows from the left table
3.
Left Employee e has 10 rows
Right Employee m has 10 rows
-- LEFT (OUTER) JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
There are 8 matching rows.
There are 2 non-matching rows from Left Employee e
There are 2 non-matching rows from Right Employee m
-->
```

```
LEFT (OUTER) JOIN will show
(8 matching rows
+ 2 non-matching rows from Left Employee e).
*/
```

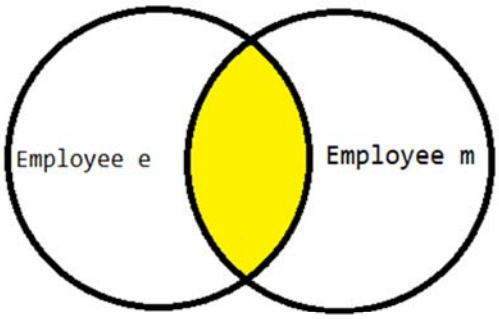



```
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) AS ManagerFullName
FROM    dbo.Employee e
        LEFT OUTER JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
             --LEFT JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;

/*
1.
Best SelfJoin_LEFT/Right (Outer) Join
2.
- LEFT (OUTER) JOIN
Returns all the matching rows +
non matching rows from the left table
3.
Left Employee e has 10 rows
Right Employee m has 10 rows
-- LEFT (OUTER) JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
There are 8 matching rows.
There are 2 non-matching rows from Left Employee e
There are 2 non-matching rows from Right Employee m
-->
LEFT (OUTER) JOIN will show
(8 matching rows
+ 2 non-matching rows from Left Employee e).
*/
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

	FullName	ManagerFullName
1	First1 Middle1 Last1	NULL
2	First2 Middle2 Last2	First1 Middle1 Last1
3	Fisrt3 Middle3 Last3	First1 Middle1 Last1
4	First4 Middle4 Last4	First2 Middle2 Last2
5	First5 Middle5 Last5	First2 Middle2 Last2
6	First6 Middle6 Last6	First2 Middle2 Last2
7	First7 Middle7 Last7	Fisrt3 Middle3 Last3
8	First8 Middle8 Last8	Fisrt3 Middle3 Last3
9	First9 Middle9 last9	Fisrt3 Middle3 Last3
10	First10 Middle10 Last10	NULL

```
--============================================================================
--T005_06_02
--2nd-Best SelfJoin_(INNER) JOIN
SELECT  *
FROM    Employee;
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) AS ManagerFullName
FROM    dbo.Employee e
        INNER JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
             --JOIN JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
GO -- Run the prvious command and begins new batch
/*
1.
2nd-Best SelfJoin_(INNER) JOIN
2.
- (INNER) JOIN
Returns only the matching rows.
Non matching rows are eliminated.
3.
Left Employee e has 10 rows
Right Employee m has 10 rows
-- LEFT (OUTER) JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
There are 8 matching rows.
There are 2 non-matching rows from Left Employee e
There are 2 non-matching rows from Right Employee m
-->
(INNER) JOIN will show
(8 matching rows)
*/
```



(Inner)Join
--> Self Join

```
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) AS ManagerFullName
FROM    dbo.Employee e
        INNER JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
             --JOIN JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;

/*
1.
2nd-Best SelfJoin_(INNER) JOIN
2.
- (INNER) JOIN
Returns only the matching rows.
Non matching rows are eliminated.
3.
Left Employee e has 10 rows
Right Employee m has 10 rows
-- LEFT (OUTER) JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
There are 8 matching rows.
There are 2 non-matching rows from Left Employee e
There are 2 non-matching rows from Right Employee m
-->
(INNER) JOIN will show
(8 matching rows)
*/
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

	FullName	ManagerFullName
1	First2 Middle2 Last2	First1 Middle1 Last1
2	Fisrt3 Middle3 Last3	First1 Middle1 Last1
3	First4 Middle4 Last4	First2 Middle2 Last2
4	First5 Middle5 Last5	First2 Middle2 Last2
5	First6 Middle6 Last6	First2 Middle2 Last2
6	First7 Middle7 Last7	Fisrt3 Middle3 Last3
7	First8 Middle8 Last8	Fisrt3 Middle3 Last3
8	First9 Middle9 last9	Fisrt3 Middle3 Last3

```
--=============================================================================
--T005_06_03
```

```sql
--Worst SelfJoin_CROSS Join - No sense
SELECT  *
FROM    Employee;
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) AS ManagerFullName
FROM    dbo.Employee e
        CROSS JOIN dbo.Employee m;
GO -- Run the prvious command and begins new batch
/*
1.
Worst SelfJoin_CROSS Join - No sense
2.
-- CROSS JOIN
Returns Cartesian product of the tables
involved in the join.
CROSS JOIN does not need ON
3.
Left Employee e has 10 rows
Right Employee m has 10 rows
-->
CROSS JOIN JOIN will show
(10 rows from Left Employee e) *
(10 rows from Right Employee m) = 100 rows
*/
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

	FullName	ManagerFullName
1	First1 Middle1 Last1	First1 Middle1 Last1
2	First2 Middle2 Last2	First1 Middle1 Last1
3	Fisrt3 Middle3 Last3	First1 Middle1 Last1
4	First4 Middle4 Last4	First1 Middle1 Last1
5	First5 Middle5 Last5	First1 Middle1 Last1
6	First6 Middle6 Last6	First1 Middle1 Last1
7	First7 Middle7 Last7	First1 Middle1 Last1
8	First8 Middle8 Last8	First1 Middle1 Last1
9	First9 Middle9 last9	First1 Middle1 Last1
10	First10 Middle10 Last10	First1 Middle1 Last1
11	First1 Middle1 Last1	First2 Middle2 Last2
12	First2 Middle2 Last2	First2 Middle2 Last2
13	Fisrt3 Middle3 Last3	First2 Middle2 Last2
14	First4 Middle4 Last4	First2 Middle2 Last2
15	First5 Middle5 Last5	First2 Middle2 Last2
16	First6 Middle6 Last6	First2 Middle2 Last2
17	First7 Middle7 Last7	First2 Middle2 Last2
18	First8 Middle8 Last8	First2 Middle2 Last2

SP1) | N550JKL\Ipmpl (52) | Sample | 00:00:00 | 100 rows

2. ISNULL(A,B) V.S. CaseWhen V.S. COALESCE

```
--=============================================================================
--T005_07_ISNULL(A,B) V.S. CaseWhen V.S. COALESCE
--=============================================================================
--=============================================================================
--T005_07_01
--Best SelfJoin_LEFT/Right (Outer) Join
SELECT  *
FROM    Employee;
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) AS ManagerFullName
FROM    dbo.Employee e
        LEFT OUTER JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
             --LEFT JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
GO -- Run the prvious command and begins new batch
/*
1.
Best SelfJoin_LEFT/Right (Outer) Join
2.
- LEFT (OUTER) JOIN
Returns all the matching rows +
non matching rows from the left table
3.
Left Employee e has 10 rows
Right Employee m has 10 rows
-- LEFT (OUTER) JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
There are 8 matching rows.
There are 2 non-matching rows from Left Employee e
There are 2 non-matching rows from Right Employee m
-->
LEFT (OUTER) JOIN will show
(8 matching rows
+ 2 non-matching rows from Left Employee e).
*/
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

	FullName	ManagerFullName
1	First1 Middle1 Last1	NULL
2	First2 Middle2 Last2	First1 Middle1 Last1
3	Fisrt3 Middle3 Last3	First1 Middle1 Last1
4	First4 Middle4 Last4	First2 Middle2 Last2
5	First5 Middle5 Last5	First2 Middle2 Last2
6	First6 Middle6 Last6	First2 Middle2 Last2
7	First7 Middle7 Last7	Fisrt3 Middle3 Last3
8	First8 Middle8 Last8	Fisrt3 Middle3 Last3
9	First9 Middle9 last9	Fisrt3 Middle3 Last3
10	First10 Middle10 Last10	NULL

```
--============================================================================
--T005_07_02
--Best SelfJoin_LEFT/Right (Outer) Join
--ISNULL(A,B)
-- if A is NULL then B, if A is not NULL then A.
SELECT  *
FROM    Employee;
SELECT  ISNULL(NULL, 'No Manager') AS ManagerFullName;
SELECT  ISNULL('Name1', 'No Manager') AS ManagerFullName;
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        ISNULL(( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ),
               'No Manager') AS ManagerFullName
        --( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) AS ManagerFullName
FROM    dbo.Employee e
        LEFT OUTER JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
             --LEFT JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
GO -- Run the prvious command and begins new batch
/*
1.
Best SelfJoin_LEFT/Right (Outer) Join
2.
- LEFT (OUTER) JOIN
Returns all the matching rows +
non matching rows from the left table
3.
Left Employee e has 10 rows
Right Employee m has 10 rows
-- LEFT (OUTER) JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
There are 8 matching rows.
There are 2 non-matching rows from Left Employee e
There are 2 non-matching rows from Right Employee m
-->
LEFT (OUTER) JOIN will show
(8 matching rows
+ 2 non-matching rows from Left Employee e).
4.
--ISNULL(A,B)
if A is NULL then B, if A is not NULL then A.
--ISNULL(( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ),
--    'No Manager') AS ManagerFullName
if ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) is NULL, then 'No Manager'
if ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) is NOT NULL, then
( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName )
*/
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

	ManagerFullName
1	No Manager

	ManagerFullName
1	Name1

	FullName	ManagerFullName
1	First1 Middle1 Last1	No Manager
2	First2 Middle2 Last2	First1 Middle1 Last1
3	Fisrt3 Middle3 Last3	First1 Middle1 Last1
4	First4 Middle4 Last4	First2 Middle2 Last2
5	First5 Middle5 Last5	First2 Middle2 Last2
6	First6 Middle6 Last6	First2 Middle2 Last2
7	First7 Middle7 Last7	Fisrt3 Middle3 Last3
8	First8 Middle8 Last8	Fisrt3 Middle3 Last3
9	First9 Middle9 last9	Fisrt3 Middle3 Last3
10	First10 Middle10 Last10	No Manager

```
--=============================================================================
--T005_07_03
--Best SelfJoin_LEFT/Right (Outer) Join
--CASE WHEN Expression THEN 'A' ELSE 'B' END
--if expression is true, then A otherwise B
SELECT  *
FROM    Employee;
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        CASE WHEN ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) IS NULL
             THEN 'No Manager'
             ELSE ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName )
        END
             --COALESCE(( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ), 'No Manager') AS
ManagerFullName
             --ISNULL(( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ), 'No Manager') AS
ManagerFullName
        --( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) AS ManagerFullName
FROM    dbo.Employee e
        LEFT OUTER JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
             --LEFT JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
GO -- Run the prvious command and begins new batch
/*
1.
Best SelfJoin_LEFT/Right (Outer) Join
```

```
2.
- LEFT (OUTER) JOIN
Returns all the matching rows +
non matching rows from the left table
3.
Left Employee e has 10 rows
Right Employee m has 10 rows
-- LEFT (OUTER) JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
There are 8 matching rows.
There are 2 non-matching rows from Left Employee e
There are 2 non-matching rows from Right Employee m
-->
LEFT (OUTER) JOIN will show
(8 matching rows
+ 2 non-matching rows from Left Employee e).
4.
--ISNULL(A,B)
if A is NULL then B, if A is not NULL then A.
--ISNULL(( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ),
--    'No Manager') AS ManagerFullName
if ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) is NULL, then 'No Manager'
if ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) is NOT NULL, then
( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName )
5.
--CASE
--     WHEN Expression
--     THEN 'A'
--     ELSE 'B'
--END
if expression is true, then A otherwise B
*/
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

	FullName	(No column name)
1	First1 Middle1 Last1	No Manager
2	First2 Middle2 Last2	First1 Middle1 Last1
3	Fisrt3 Middle3 Last3	First1 Middle1 Last1
4	First4 Middle4 Last4	First2 Middle2 Last2
5	First5 Middle5 Last5	First2 Middle2 Last2
6	First6 Middle6 Last6	First2 Middle2 Last2
7	First7 Middle7 Last7	Fisrt3 Middle3 Last3
8	First8 Middle8 Last8	Fisrt3 Middle3 Last3
9	First9 Middle9 last9	Fisrt3 Middle3 Last3
10	First10 Middle10 Last10	No Manager

```
--=============================================================================
--T005_07_04
--Best SelfJoin_LEFT/Right (Outer) Join
```

```
--COALESCE(A, B, C ...etc)
--Return the first non-NULL value.
SELECT  *
FROM    Employee;
SELECT  ISNULL(NULL, 'No Manager') AS ManagerFullName;
SELECT  ISNULL('Name1', 'No Manager') AS ManagerFullName;
SELECT  COALESCE(NULL, 'No Manager') AS ManagerFullName;
SELECT  COALESCE('Name1', 'No Manager') AS ManagerFullName;
SELECT  ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS FullName ,
        COALESCE(( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ),
                 'No Manager') AS ManagerFullName
        --CASE WHEN ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) IS NULL
        --     THEN 'No Manager'
        --     ELSE ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName )
        --END
             --ISNULL(( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ), 'No Manager') AS
ManagerFullName
        --( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) AS ManagerFullName
FROM    dbo.Employee e
        LEFT OUTER JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
             --LEFT JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
GO -- Run the prvious command and begins new batch
/*
1.
Best SelfJoin_LEFT/Right (Outer) Join
2.
- LEFT (OUTER) JOIN
Returns all the matching rows +
non matching rows from the left table
3.
Left Employee e has 10 rows
Right Employee m has 10 rows
-- LEFT (OUTER) JOIN dbo.Employee m ON e.ReportsTo = m.EmployeeID;
There are 8 matching rows.
There are 2 non-matching rows from Left Employee e
There are 2 non-matching rows from Right Employee m
-->
LEFT (OUTER) JOIN will show
(8 matching rows
+ 2 non-matching rows from Left Employee e).
4.
--ISNULL(A,B)
if A is NULL then B, if A is not NULL then A.
--ISNULL(( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ),
--    'No Manager') AS ManagerFullName
if ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) is NULL, then 'No Manager'
if ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ) is NOT NULL, then
( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName )
5.
--CASE
--     WHEN Expression
--     THEN 'A'
--     ELSE 'B'
--END
if expression is true, then A otherwise B
6.
--COALESCE(A, B, C ...etc)
Return the first non-NULL value.
--COALESCE(( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName ),
--    'No Manager') AS ManagerFullName
first value is ( m.FirstName + ' ' + m.MiddleName + ' ' + m.LastName )
second value is  'No Manager'
Return the first non-NULL value.
```

```
*/
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

	ManagerFullName
1	No Manager

	ManagerFullName
1	Name1

	ManagerFullName
1	No Manager

	ManagerFullName
1	Name1

	FullName	ManagerFullName
1	First1 Middle1 Last1	No Manager
2	First2 Middle2 Last2	First1 Middle1 Last1
3	Fisrt3 Middle3 Last3	First1 Middle1 Last1
4	First4 Middle4 Last4	First2 Middle2 Last2
5	First5 Middle5 Last5	First2 Middle2 Last2
6	First6 Middle6 Last6	First2 Middle2 Last2
7	First7 Middle7 Last7	Fisrt3 Middle3 Last3
8	First8 Middle8 Last8	Fisrt3 Middle3 Last3
9	First9 Middle9 last9	Fisrt3 Middle3 Last3
10	First10 Middle10 Last10	No Manager

```
--=============================================================================
--T005_07_05
--COALESCE(A, B, C ...etc)
--Return the first non-NULL value.
SELECT  *
FROM    Player;
SELECT  COALESCE(NULL, 'A', 'B') AS FirstAvailableLeader;
SELECT  COALESCE(NULL, NULL, 'B', 'A') AS FirstAvailableLeader;
SELECT  COALESCE('D', NULL, 'B', 'C') AS FirstAvailableLeader;
SELECT  p.PlayerID ,
        p.Name ,
        COALESCE(p.FirstLeaderID, p.SecondLeaderID, p.ThirdLeaderID, 0) AS FirstAvailableLeaderID ,
        (
                    --SELECT ISNULL(p2.[Name], 'No Leader')
          SELECT    p2.[Name]
          FROM      dbo.Player p2
```

```
            WHERE       p2.PlayerID = COALESCE(p.FirstLeaderID, p.SecondLeaderID,
                                               p.ThirdLeaderID)
        ) AS FirstAvailableLeader
FROM    dbo.Player p;
GO -- Run the prvious command and begins new batch
/*
1.
--COALESCE(A, B, C ...etc)
Return the first non-NULL value.
2.
--SELECT  COALESCE(NULL, 'A', 'B') AS FirstAvailableLeader;
Return A
--SELECT  COALESCE(NULL, NULL, 'B', 'A') AS FirstAvailableLeader;
Return B
--SELECT  COALESCE('D', NULL, 'B', 'C') AS FirstAvailableLeader;
Return D
3.
-- COALESCE(p.FirstLeaderID, p.SecondLeaderID, p.ThirdLeaderID, 0) AS FirstAvailableLeaderID ,
first valus is FirstLeaderID
second value is SecondLeaderID
third value is ThirdLeaderID
fourth value is 0
Return the first non-NULL value
*/
```

	PlayerID	Name	FirstLeaderID	SecondLeaderID	ThirdLeaderID
1	1	Name1	NULL	NULL	NULL
2	2	Name2	NULL	1	NULL
3	3	Name3	NULL	1	2
4	4	Name4	1	2	3
5	5	Name5	NULL	NULL	1
6	6	Name6	NULL	2	3
7	7	Name7	NULL	NULL	3
8	8	Name8	NULL	1	2
9	9	Name9	1	2	3
10	10	Name10	NULL	1	2

	FirstAvailableLeader
1	A

	FirstAvailableLeader
1	B

	FirstAvailableLeader
1	D

	PlayerID	Name	FirstAvailableLeaderID	FirstAvailableLeader
1	1	Name1	0	NULL
2	2	Name2	1	Name1
3	3	Name3	1	Name1
4	4	Name4	1	Name1
5	5	Name5	1	Name1
6	6	Name6	2	Name2
7	7	Name7	3	Name3
8	8	Name8	1	Name1
9	9	Name9	1	Name1
10	10	Name10	1	Name1

==========

3. Union(All)

```
--============================================================================
--T005_08_Union(All)
--============================================================================
--============================================================================
--T005_08_01
--UNION ALL
SELECT  *
FROM    dbo.Person;
SELECT  *
FROM    dbo.Person2;
SELECT  p1.[Name] ,
        p1.[Email]
FROM    dbo.Person p1
UNION ALL
SELECT  p2.[Name] ,
        p2.[Email]
FROM    dbo.Person2 p2;
GO -- Run the prvious command and begins new batch
/*
Person has 10 rows
Person2 has 10 rows
Compare Person and Person2, there are 5 rows are the same.
-->
UNION removes duplicate rows,
UNION ALL does not.
-->
UNION ALL
-->
(10 rows from Person
+ 10 rows from Person2) = 20 rows
*/
```

	PersonID	Name	Email	FirstLeaderID	SecondLeaderID	ThirdLeaderID
1	1	Name1	1@1.com	NULL	NULL	NULL
2	2	Name2	2@2.com	NULL	1	NULL
3	3	Name3	3@3.com	NULL	1	2
4	4	Name4	4@4.com	1	2	3
5	5	Name5	5@5.com	NULL	NULL	1
6	6	Name6	6@6.com	NULL	2	3
7	7	Name7	7@7.com	NULL	NULL	3
8	8	Name8	8@8.com	NULL	1	2
9	9	Name9	9@9.com	1	2	3
10	10	Name10	10@10.com	NULL	1	2

	Person2ID	Name	Email
1	1	Name6	6@6.com
2	2	Name7	7@7.com
3	3	Name8	8@8.com
4	4	Name9	9@9.com
5	5	Name10	10@10.com
6	6	Name11	11@11.com
7	7	Name12	12@12.com
8	8	Name13	13@13.com
9	9	Name14	14@14.com
10	10	Name15	15@15.com

	Name	Email
1	Name1	1@1.com
2	Name2	2@2.com
3	Name3	3@3.com
4	Name4	4@4.com
5	Name5	5@5.com
6	Name6	6@6.com
7	Name7	7@7.com
8	Name8	8@8.com
9	Name9	9@9.com
10	Name10	10@10.com
11	Name6	6@6.com
12	Name7	7@7.com
13	Name8	8@8.com
14	Name9	9@9.com
15	Name10	10@10.com
16	Name11	11@11.com
17	Name12	12@12.com
18	Name13	13@13.com
19	Name14	14@14.com
20	Name15	15@15.com

```
--============================================================================
--T005_08_02
--UNION
SELECT  *
FROM    dbo.Person;
SELECT  *
FROM    dbo.Person2;
SELECT  p1.[Name] ,
        p1.[Email]
FROM    dbo.Person p1
UNION
SELECT  p2.[Name] ,
        p2.[Email]
FROM    dbo.Person2 p2;
GO -- Run the prvious command and begins new batch
/*
Person has 10 rows
```

```
Person2 has 10 rows
Compare Person and Person2, there are 5 rows are the same.
-->
UNION removes duplicate rows,
UNION ALL does not.
-->
UNION
-->
(10 rows from Person
+ 10 rows from Person2
- 5 duplicate rows) = 15 rows
*/
```

	PersonID	Name	Email	FirstLeaderID	SecondLeaderID	ThirdLeaderID
1	1	Name1	1@1.com	NULL	NULL	NULL
2	2	Name2	2@2.com	NULL	1	NULL
3	3	Name3	3@3.com	NULL	1	2
4	4	Name4	4@4.com	1	2	3
5	5	Name5	5@5.com	NULL	NULL	1
6	6	Name6	6@6.com	NULL	2	3
7	7	Name7	7@7.com	NULL	NULL	3
8	8	Name8	8@8.com	NULL	1	2
9	9	Name9	9@9.com	1	2	3
10	10	Name10	10@10.com	NULL	1	2

	Person2ID	Name	Email
1	1	Name6	6@6.com
2	2	Name7	7@7.com
3	3	Name8	8@8.com
4	4	Name9	9@9.com
5	5	Name10	10@10.com
6	6	Name11	11@11.com
7	7	Name12	12@12.com
8	8	Name13	13@13.com
9	9	Name14	14@14.com
10	10	Name15	15@15.com

	Name	Email
1	Name1	1@1.com
2	Name10	10@10.com
3	Name11	11@11.com
4	Name12	12@12.com
5	Name13	13@13.com
6	Name14	14@14.com
7	Name15	15@15.com
8	Name2	2@2.com
9	Name3	3@3.com
10	Name4	4@4.com
11	Name5	5@5.com
12	Name6	6@6.com
13	Name7	7@7.com
14	Name8	8@8.com
15	Name9	9@9.com

```sql
--============================================================================
--T005_08_03
--UNION(ALL)...OrderBy
SELECT  *
FROM    dbo.Employee;
SELECT  *
FROM    dbo.Employee
WHERE   DepartmentID = 1
--ORDER BY FirstName;   --ERROR
UNION ALL
SELECT  *
FROM    dbo.Employee
WHERE   DepartmentID = 2
--ORDER BY FirstName;   --ERROR
UNION ALL
SELECT  *
FROM    dbo.Employee
WHERE   DepartmentID = 3
ORDER BY DepartmentID;
GO -- Run the prvious command and begins new batch
/*
1.
DepartmentID = 1  has 3 rows
DepartmentID = 2  has 3 rows
DepartmentID = 3  has 2 rows
-->
UNION removes duplicate rows,
UNION ALL does not.
-->
UNION ALL
-->
(3 rows from DepartmentID = 1
+ 3 rows from DepartmentID = 2
+ 2 rows from DepartmentID = 3) = 8 rows
2.
ORDER BY caluse can only be used on the last SELECT statement.
ORDER BY caluse is on any other SELECT statement will cause Syntax Error.
UNION combines rows from 2 or more tables/Search Results.
Thus, ORDER BY caluse can only be used after all the results is combined.
3.
UNION combines rows from 2 or more tables/Search Results.
JOINS combine columns from 2 or more tables.
*/
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	4	2	First4	Middle4	Last4	1
3	7	3	First7	Middle7	Last7	1
4	2	1	First2	Middle2	Last2	2
5	5	2	First5	Middle5	Last5	2
6	8	3	First8	Middle8	Last8	2
7	3	1	Fisrt3	Middle3	Last3	3
8	6	2	First6	Middle6	Last6	3

```
--=============================================================================
--T005_08_04
--Different between UNION(ALL) and JOINS
SELECT  *
FROM    dbo.Employee;
SELECT  *
FROM    dbo.Department;
SELECT  e.EmployeeID ,
        ( e.FirstName + ' ' + e.MiddleName + ' ' + e.LastName ) AS [Name] ,
        d.DepartmentID ,
        d.DepartmentName
FROM    dbo.Employee e
        INNER JOIN dbo.Department d ON e.DepartmentID = d.DepartmentID
ORDER BY d.DepartmentID;
GO -- Run the prvious command and begins new batch
/*
UNION (ALL) combines rows from 2 or more tables/Search Results.
JOINS combine columns from 2 or more tables.
*/
```

	EmployeeID	ReportsTo	FirstName	MiddleName	LastName	DepartmentID
1	1	NULL	First1	Middle1	Last1	1
2	2	1	First2	Middle2	Last2	2
3	3	1	Fisrt3	Middle3	Last3	3
4	4	2	First4	Middle4	Last4	1
5	5	2	First5	Middle5	Last5	2
6	6	2	First6	Middle6	Last6	3
7	7	3	First7	Middle7	Last7	1
8	8	3	First8	Middle8	Last8	2
9	9	3	First9	Middle9	last9	NULL
10	10	NULL	First10	Middle10	Last10	NULL

	DepartmentID	DepartmentName
1	1	Department1
2	2	Department2
3	3	Department3
4	4	Department4
5	5	Department5
6	6	Department6

	EmployeeID	Name	DepartmentID	DepartmentName
1	1	First1 Middle1 Last1	1	Department1
2	4	First4 Middle4 Last4	1	Department1
3	7	First7 Middle7 Last7	1	Department1
4	8	First8 Middle8 Last8	2	Department2
5	5	First5 Middle5 Last5	2	Department2
6	2	First2 Middle2 Last2	2	Department2
7	3	Fisrt3 Middle3 Last3	3	Department3
8	6	First6 Middle6 Last6	3	Department3

```
--=============================================================================
--T005_09_Clean up
--=============================================================================
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE      TABLE_NAME = 'Employee' ) )
    BEGIN
        TRUNCATE TABLE Employee;
        DROP TABLE Employee;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE      TABLE_NAME = 'Department' ) )
    BEGIN
        TRUNCATE TABLE Department;
        DROP TABLE Department;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE      TABLE_NAME = 'CarModel' ) )
    BEGIN
        TRUNCATE TABLE CarModel;
        DROP TABLE CarModel;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE      TABLE_NAME = 'CarColor' ) )
    BEGIN
        TRUNCATE TABLE CarColor;
        DROP TABLE CarColor;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
```

```
            FROM      INFORMATION_SCHEMA.TABLES
            WHERE     TABLE_NAME = 'Player' ) )
    BEGIN
        TRUNCATE TABLE Player;
        DROP TABLE Player;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
            FROM      INFORMATION_SCHEMA.TABLES
            WHERE     TABLE_NAME = 'Person' ) )
    BEGIN
        TRUNCATE TABLE Person;
        DROP TABLE Person;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
            FROM      INFORMATION_SCHEMA.TABLES
            WHERE     TABLE_NAME = 'Person2' ) )
    BEGIN
        TRUNCATE TABLE Person2;
        DROP TABLE Person2;
    END;
GO -- Run the previous command and begins new batch
```