(T27)深入理解 Join 中的 Except 和 NotIn 和 Insert 和 Union 和 UnionAll。比較 InnerJoin 和 DistinctInnerJoin

CourseGUID: e48417fc-9db5-4e99-822c-706c5ccef6cc

==

(T27)深入理解 Join 中的 Except 和 NotIn 和 Insert 和 Union 和 UnionAll。比較 InnerJoin 和 DistinctInnerJoin

==

==

0. Summary

1.
1.1.
--Except, INTERSECT, UNION, UNION ALL
Except, INTERSECT, UNION, UNION ALL operators deal with rows, not columns.
In order to use Except, INTERSECT, UNION, UNION ALL,
the order and the the number of the columns from the select cluase
must be the same as all queries.
The data types must be same or at least compatible as all queries.
1.1.1
--UNION
UNION operator returns "DISTINCT rows" from both QueryA and QueryB.
1.1.2.
--UNION ALL
UNION ALL operator returns all rows from both QueryA and QueryB,
and it may "contains duplicates rows"
1.1.3.
--INTERSECT
INTERSECT operator retrieves the "DISTINCT rows"
which exists in both QueryA and QueryB,

1.1.4.
--EXCEPT
1.1.4.1.
--QueryA EXCEPT QueryB
EXCEPT operator retrieves the "DISTINCT rows" from PersonA
that does not exist in PersonB
1.1.4.2.
--QueryB EXCEPT QueryA
EXCEPT operator retrieves the "DISTINCT rows" from PersonB
that does not exist in PersonA

1.2.
--INNER JOIN, DISTINCT INNER JOIN
INNER JOIN, DISTINCT INNER JOIN operators deal with columns
by using JoinColumns.
1.2.1.
--INTERSECT V.S. INNER JOIN V.S. DISTINCT INNER JOIN
1.2.1.1.
INTERSECT and DISTINCT INNER JOIN both return "DISTINCT rows".
but INNER JOIN may return duplicated rows.
1.2.1.2.
If columnA INNER JOIN columnB,
and If value of columnARow5 is NULL,
and If value of columnBRow5 is NULL,
1.2.1.2.1.
Then INTERSECT treats two NULLs as a same value
and it will think columnARow5 and columnBRow5 are matching row.
It will display this matching row.
1.2.1.2.2.
Then (DISTINCT) INNER JOIN treats two NULLs as a different value
and it will think columnARow5 and columnBRow5 are NOT matching row.
It will NOT display this row.

1.3.
--NOT IN
NOT IN compares ONE column from the Outer query
with a ONE column from the Inner query.
NOT IN get the rows from the outter query
that aren't in the Inner query's results.
NOT IN might "contain duplicated rows".
1.3.1.
--Except V.S. NOT IN
Except and NOT IN both get the rows from the left/outter query
that aren't in the right/Inner query's results.
Except returns "DISTINCT ROWS",
but NOT IN may return "duplicated rows".
1.3.2.
--Except
1.3.2.1.
Except, INTERSECT, UNION, UNION ALL operators deal with rows, not columns.
In order to use Except, INTERSECT, UNION, UNION ALL,
the order and the the number of the columns from the select cluase
must be the same as all queries.
The data types must be same or at least compatible as all queries.
1.3.2.2.
Except operator only get the rows from the left query
that aren't in the right query's results.
Except will return "DISTINCT ROWS".
1.3.3.
--NOT IN
NOT IN compares ONE column from the Outer query
with a ONE column from the Inner query.
NOT IN may return "duplicated rows".

==

1. Except V.S. NOT IN

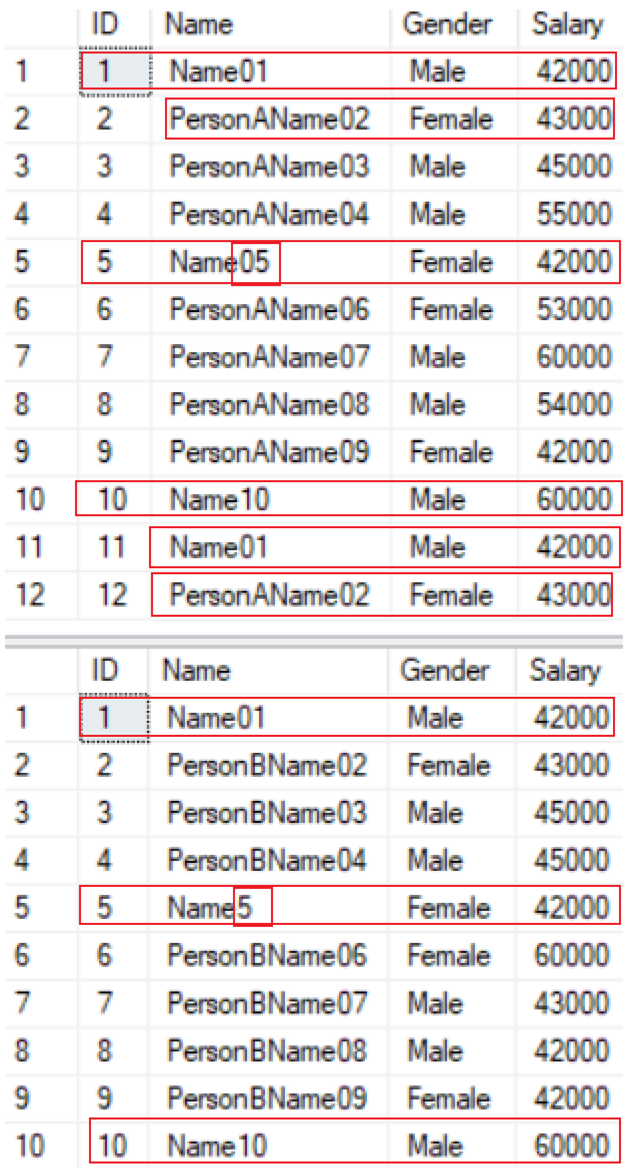
```
--==================================================================
--T027_01_Except V.S. NOT IN
--==================================================================
/*
1.
Except V.S. NOT IN
1.1.
Except
1.1.1.
In order to use Except,
the order and the the number of the columns from the select cluase
must be the same as all queries.
1.1.2.
Except operator only get the "DISTINCT ROWS" from the left query
that aren’t in the right query’s results.
1.2.
NOT IN
1.2.1.
NOT IN compares ONE column from the outer query with a ONE column from the subquery.
1.2.2.
NOT IN does "NOT FILTER OUT DUPLICATED" rows in the result.
*/
```

1.1. Create Sample Data

```
--==================================================================
--T027_01_01
--Create Sample Data
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'PersonA' ) )
    BEGIN
        TRUNCATE TABLE dbo.PersonA;
        DROP TABLE PersonA;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'PersonB' ) )
    BEGIN
        TRUNCATE TABLE dbo.PersonB;
        DROP TABLE PersonB;
    END;
GO -- Run the previous command and begins new batch
CREATE TABLE PersonA
    (
      ID INT IDENTITY(1, 1)
             PRIMARY KEY ,
      [Name] NVARCHAR(100) ,
      Gender NVARCHAR(10) ,
      Salary INT
    );
GO -- Run the previous command and begins new batch
INSERT  INTO PersonA
VALUES  ( 'Name01', 'Male', 42000 );
```

```sql
INSERT  INTO PersonA
VALUES  ( 'PersonAName02', 'Female', 43000 );
INSERT  INTO PersonA
VALUES  ( 'PersonAName03', 'Male', 45000 );
INSERT  INTO PersonA
VALUES  ( 'PersonAName04', 'Male', 55000 );
INSERT  INTO PersonA
VALUES  ( 'Name05', 'Female', 42000 );
INSERT  INTO PersonA
VALUES  ( 'PersonAName06', 'Female', 53000 );
INSERT  INTO PersonA
VALUES  ( 'PersonAName07', 'Male', 60000 );
INSERT  INTO PersonA
VALUES  ( 'PersonAName08', 'Male', 54000 );
INSERT  INTO PersonA
VALUES  ( 'PersonAName09', 'Female', 42000 );
INSERT  INTO PersonA
VALUES  ( 'Name10', 'Male', 60000 );
--The duplicated Rows
INSERT  INTO PersonA
VALUES  ( 'Name01', 'Male', 42000 );
INSERT  INTO PersonA
VALUES  ( 'PersonAName02', 'Female', 43000 );
GO -- Run the previous command and begins new batch
CREATE TABLE PersonB
    (
      ID INT IDENTITY(1, 1)
             PRIMARY KEY ,
      [Name] NVARCHAR(100) ,
      Gender NVARCHAR(10) ,
      Salary INT
    );
GO -- Run the previous command and begins new batch
INSERT  INTO PersonB
VALUES  ( 'Name01', 'Male', 42000 );
INSERT  INTO PersonB
VALUES  ( 'PersonBName02', 'Female', 43000 );
INSERT  INTO PersonB
VALUES  ( 'PersonBName03', 'Male', 45000 );
INSERT  INTO PersonB
VALUES  ( 'PersonBName04', 'Male', 45000 );
INSERT  INTO PersonB
VALUES  ( 'Name5', 'Female', 42000 );
INSERT  INTO PersonB
VALUES  ( 'PersonBName06', 'Female', 60000 );
INSERT  INTO PersonB
VALUES  ( 'PersonBName07', 'Male', 43000 );
INSERT  INTO PersonB
VALUES  ( 'PersonBName08', 'Male', 42000 );
INSERT  INTO PersonB
VALUES  ( 'PersonBName09', 'Female', 42000 );
INSERT  INTO PersonB
VALUES  ( 'Name10', 'Male', 60000 );
```

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



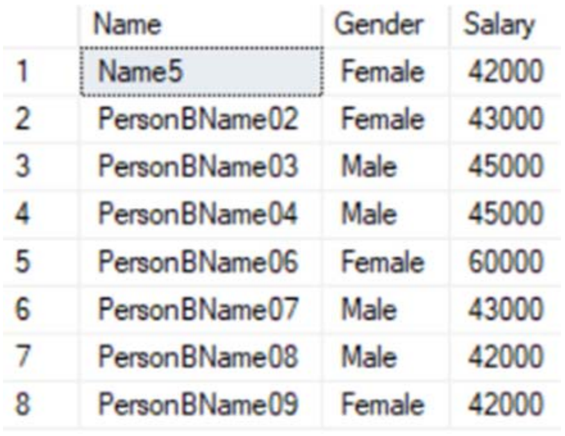
	ID	Name	Gender	Salary
1	1	Name01	Male	42000
2	2	PersonAName02	Female	43000
3	3	PersonAName03	Male	45000
4	4	PersonAName04	Male	55000
5	5	Name05	Female	42000
6	6	PersonAName06	Female	53000
7	7	PersonAName07	Male	60000
8	8	PersonAName08	Male	54000
9	9	PersonAName09	Female	42000
10	10	Name10	Male	60000
11	11	Name01	Male	42000
12	12	PersonAName02	Female	43000

	ID	Name	Gender	Salary
1	1	Name01	Male	42000
2	2	PersonBName02	Female	43000
3	3	PersonBName03	Male	45000
4	4	PersonBName04	Male	45000
5	5	Name5	Female	42000
6	6	PersonBName06	Female	60000
7	7	PersonBName07	Male	43000
8	8	PersonBName08	Male	42000
9	9	PersonBName09	Female	42000
10	10	Name10	Male	60000

1.2. Except V.S. NOT IN ; Except for 2 tables

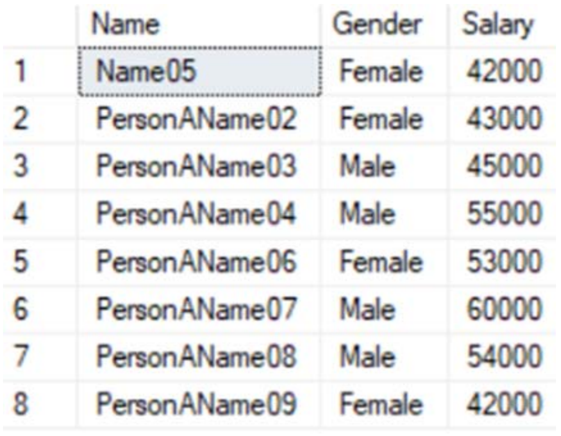
```
--======================================================================
--T027_01_02
--Except V.S. NOT IN ; Except for 2 tables
------------------------------------------------------------------
--T027_01_02_01
--select all rows from TableB that does not exist in TableA.
SELECT  [Name] ,
        Gender ,
        Salary
FROM    dbo.PersonB
EXCEPT
SELECT  [Name] ,
        Gender ,
        Salary
FROM    dbo.PersonA;
```

```
GO -- Run the previous command and begins new batch
```



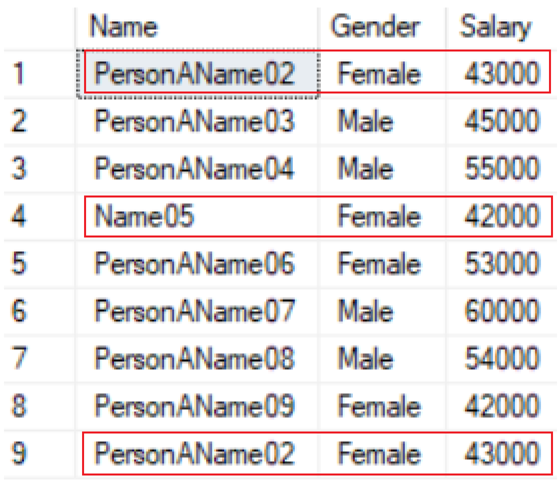
	Name	Gender	Salary
1	Name5	Female	42000
2	PersonBName02	Female	43000
3	PersonBName03	Male	45000
4	PersonBName04	Male	45000
5	PersonBName06	Female	60000
6	PersonBName07	Male	43000
7	PersonBName08	Male	42000
8	PersonBName09	Female	42000

```
-----------------------------------------------------------------
--T027_01_02_02
--select all rows from TableA that does not exist in TableB.
SELECT  [Name] ,
        Gender ,
        Salary
FROM    dbo.PersonA
EXCEPT
SELECT  [Name] ,
        Gender ,
        Salary
FROM    dbo.PersonB;
GO -- Run the previous command and begins new batch
```



	Name	Gender	Salary
1	Name05	Female	42000
2	PersonAName02	Female	43000
3	PersonAName03	Male	45000
4	PersonAName04	Male	55000
5	PersonAName06	Female	53000
6	PersonAName07	Male	60000
7	PersonAName08	Male	54000
8	PersonAName09	Female	42000

```
-----------------------------------------------------------------
--T027_01_02_03
--select all rows from TableA that does not exist in TableB.
SELECT  [Name] ,
        Gender ,
        Salary
FROM    dbo.PersonA
WHERE   [Name] NOT IN ( SELECT  [Name]
                        FROM    dbo.PersonB );
GO -- Run the previous command and begins new batch
```

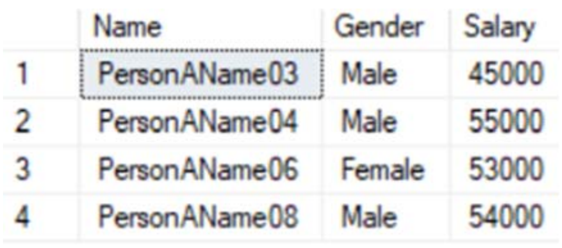


	Name	Gender	Salary
1	PersonAName02	Female	43000
2	PersonAName03	Male	45000
3	PersonAName04	Male	55000
4	Name05	Female	42000
5	PersonAName06	Female	53000
6	PersonAName07	Male	60000
7	PersonAName08	Male	54000
8	PersonAName09	Female	42000
9	PersonAName02	Female	43000

```
/*
1.
Except V.S. NOT IN
1.1.
Except
1.1.1.
In order to use Except,
the order and the the number of the columns from the select cluase
must be the same as all queries.
1.1.2.
Except operator only get the "DISTINCT ROWS" from the left query
that aren’t in the right query’s results.
1.2.
NOT IN
1.2.1.
NOT IN compares ONE column from the outer query with a ONE column from the subquery.
1.2.2.
NOT IN does "NOT FILTER OUT DUPLICATED" rows in the result.
*/
```

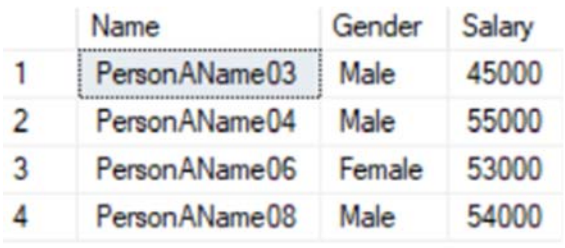
1.3. Except for 1 table

```
--======================================================================
--T027_01_03
--Except for 1 table
--------------------------------------------------------------------
--T027_01_03_01
--Salary >= 45000 AND Salary <= 58000
SELECT  [Name] ,
        Gender ,
        Salary
FROM    dbo.PersonA
WHERE   Salary >= 45000
EXCEPT
SELECT  [Name] ,
        Gender ,
        Salary
FROM    dbo.PersonA
WHERE   Salary >= 58000
ORDER BY [Name];
GO -- Run the previous command and begins new batch
```



	Name	Gender	Salary
1	PersonAName03	Male	45000
2	PersonAName04	Male	55000
3	PersonAName06	Female	53000
4	PersonAName08	Male	54000

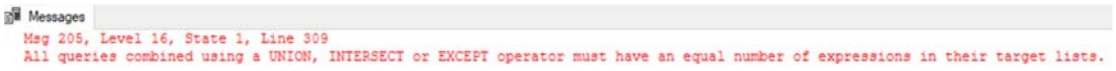
```
/*
The result is same as
--WHERE   Salary >= 45000 AND Salary <= 58000
*/
-------------------------------------------------------------------
--T027_01_03_02
--Salary >= 45000 AND Salary <= 58000
SELECT  [Name] ,
        Gender ,
        Salary
FROM    dbo.PersonA
WHERE   Salary >= 45000
        AND Salary <= 58000;
GO -- Run the previous command and begins new batch
```



	Name	Gender	Salary
1	PersonAName03	Male	45000
2	PersonAName04	Male	55000
3	PersonAName06	Female	53000
4	PersonAName08	Male	54000

1.4. EXCEPT limit

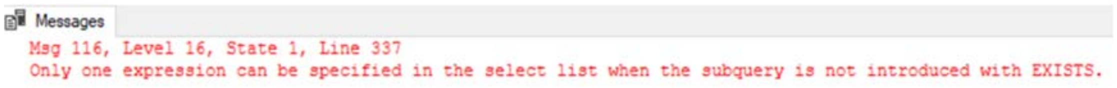
```
--=======================================================================
--T027_01_04
--EXCEPT limit
SELECT  [Name] ,
        Gender ,
        Salary
FROM    dbo.PersonA
EXCEPT
SELECT  [Name] ,
        Gender
FROM    dbo.PersonB;
/*
1.
Output
--Msg 205, Level 16, State 1, Line 250
--All queries combined using a UNION, INTERSECT or EXCEPT operator
--must have an equal number of expressions in their target lists.
In order to use Except,
the order and the the number of the columns from the select cluase
must be the same as all queries.
*/
```



```
Msg 205, Level 16, State 1, Line 309
All queries combined using a UNION, INTERSECT or EXCEPT operator must have an equal number of expressions in their target lists.
```

1.5. NOT IN limit

```
--=======================================================================
--T027_01_05
--NOT IN limit
SELECT  [Name] ,
        Gender ,
        Salary
FROM    PersonA
WHERE   ID NOT IN ( SELECT  [Name] ,
                             Gender
                   FROM     PersonB );
/*
Output
--Msg 116, Level 16, State 1, Line 274
--Only one expression can be specified in the select list
--when the subquery is not introduced with EXISTS.
NOT IN compares ONE column from the outer query
with a ONE column from the subquery.
*/
```



```
Msg 116, Level 16, State 1, Line 337
Only one expression can be specified in the select list when the subquery is not introduced with EXISTS.
```

1.6. Clean up

```
--======================================================================
--T027_01_06
--Clean up
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'PersonA' ) )
   BEGIN
       TRUNCATE TABLE dbo.PersonA;
       DROP TABLE PersonA;
   END;
GO -- Run the previous command and begins new batch
--If Table exists then DROP it
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'PersonB' ) )
   BEGIN
       TRUNCATE TABLE dbo.PersonB;
       DROP TABLE PersonB;
   END;
GO -- Run the previous command and begins new batch
==================================================================
```

2. Intersect

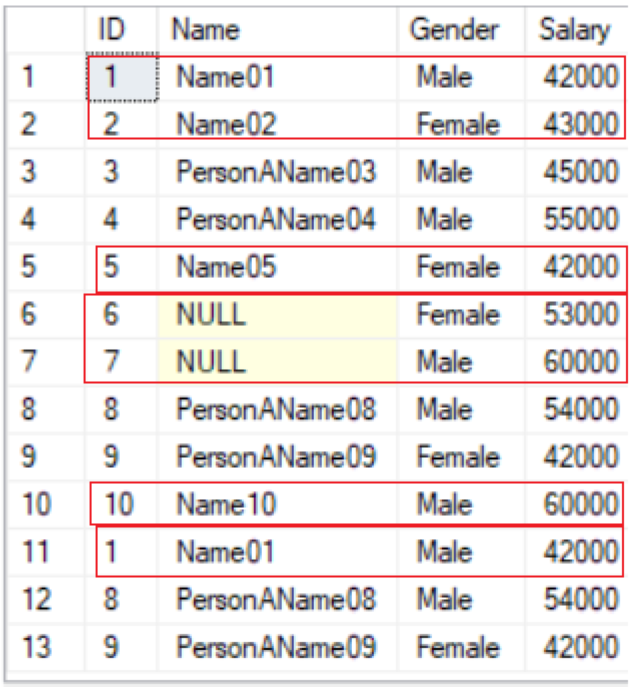
```
--======================================================================
--T027_02_Intersect
--======================================================================
/*
1.2.
--INNER JOIN, DISTINCT INNER JOIN
INNER JOIN, DISTINCT INNER JOIN operators deal with columns
by using JoinColumns.
1.2.1.
--INTERSECT  V.S. INNER JOIN  V.S.  DISTINCT INNER JOIN
1.2.1.1.
INTERSECT and DISTINCT INNER JOIN both return "DISTINCT rows".
but INNER JOIN may return duplicated rows.
1.2.1.2.
If columnA INNER JOIN columnB,
and If value of columnARow5 is NULL,
and If value of columnBRow5 is NULL,
1.2.1.2.1.
```

```
Then INTERSECT treats two NULLs as a same value
and it will think columnARow5 and columnBRow5 are matching row.
It will display this matching row.
1.2.1.2.2.
Then (DISTINCT) INNER JOIN treats two NULLs as a different value
and it will think columnARow5 and columnBRow5 are NOT matching row.
*/
```

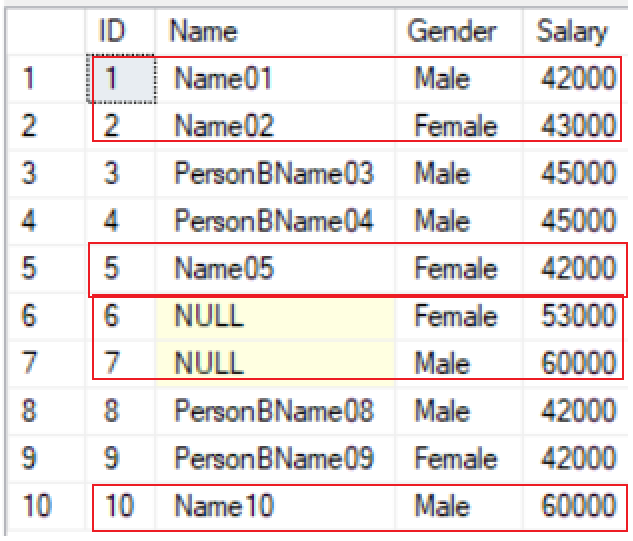
2.1. Create sample data

```
--======================================================================
--T027_02_01
--Create sample data
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'PersonA' ) )
    BEGIN
        TRUNCATE TABLE dbo.PersonA;
        DROP TABLE PersonA;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'PersonB' ) )
    BEGIN
        TRUNCATE TABLE dbo.PersonB;
        DROP TABLE PersonB;
    END;
GO -- Run the previous command and begins new batch
CREATE TABLE PersonA
    (
      ID INT ,
      [Name] NVARCHAR(100) ,
      Gender NVARCHAR(10) ,
      Salary INT
    );
GO -- Run the previous command and begins new batch
INSERT  INTO PersonA
VALUES  ( 1, 'Name01', 'Male', 42000 );
INSERT  INTO PersonA
VALUES  ( 2, 'Name02', 'Female', 43000 );
INSERT  INTO PersonA
VALUES  ( 3, 'PersonAName03', 'Male', 45000 );
INSERT  INTO PersonA
VALUES  ( 4, 'PersonAName04', 'Male', 55000 );
INSERT  INTO PersonA
VALUES  ( 5, 'Name05', 'Female', 42000 );
INSERT  INTO PersonA
VALUES  ( 6, NULL, 'Female', 53000 );
INSERT  INTO PersonA
VALUES  ( 7, NULL, 'Male', 60000 );
INSERT  INTO PersonA
VALUES  ( 8, 'PersonAName08', 'Male', 54000 );
INSERT  INTO PersonA
VALUES  ( 9, 'PersonAName09', 'Female', 42000 );
INSERT  INTO PersonA
```

```
VALUES  ( 10, 'Name10', 'Male', 60000 );
--The duplicated Rows
INSERT  INTO PersonA
VALUES  ( 1, 'Name01', 'Male', 42000 );
INSERT  INTO PersonA
VALUES  ( 8, 'PersonAName08', 'Male', 54000 );
INSERT  INTO PersonA
VALUES  ( 9, 'PersonAName09', 'Female', 42000 );
GO -- Run the previous command and begins new batch
CREATE TABLE PersonB
    (
      ID INT ,
      [Name] NVARCHAR(100) ,
      Gender NVARCHAR(10) ,
      Salary INT
    );
GO -- Run the previous command and begins new batch
INSERT  INTO PersonB
VALUES  ( 1, 'Name01', 'Male', 42000 );
INSERT  INTO PersonB
VALUES  ( 2, 'Name02', 'Female', 43000 );
INSERT  INTO PersonB
VALUES  ( 3, 'PersonBName03', 'Male', 45000 );
INSERT  INTO PersonB
VALUES  ( 4, 'PersonBName04', 'Male', 45000 );
INSERT  INTO PersonB
VALUES  ( 5, 'Name05', 'Female', 42000 );
INSERT  INTO PersonB
VALUES  ( 6, NULL, 'Female', 53000 );
INSERT  INTO PersonB
VALUES  ( 7, NULL, 'Male', 60000 );
INSERT  INTO PersonB
VALUES  ( 8, 'PersonBName08', 'Male', 42000 );
INSERT  INTO PersonB
VALUES  ( 9, 'PersonBName09', 'Female', 42000 );
INSERT  INTO PersonB
VALUES  ( 10, 'Name10', 'Male', 60000 );
GO -- Run the previous command and begins new batch
SELECT  *
FROM    dbo.PersonA;
SELECT  *
FROM    dbo.PersonB;
GO -- Run the previous command and begins new batch
```



	ID	Name	Gender	Salary
1	1	Name01	Male	42000
2	2	Name02	Female	43000
3	3	PersonAName03	Male	45000
4	4	PersonAName04	Male	55000
5	5	Name05	Female	42000
6	6	NULL	Female	53000
7	7	NULL	Male	60000
8	8	PersonAName08	Male	54000
9	9	PersonAName09	Female	42000
10	10	Name10	Male	60000
11	1	Name01	Male	42000
12	8	PersonAName08	Male	54000
13	9	PersonAName09	Female	42000



	ID	Name	Gender	Salary
1	1	Name01	Male	42000
2	2	Name02	Female	43000
3	3	PersonBName03	Male	45000
4	4	PersonBName04	Male	45000
5	5	Name05	Female	42000
6	6	NULL	Female	53000
7	7	NULL	Male	60000
8	8	PersonBName08	Male	42000
9	9	PersonBName09	Female	42000
10	10	Name10	Male	60000

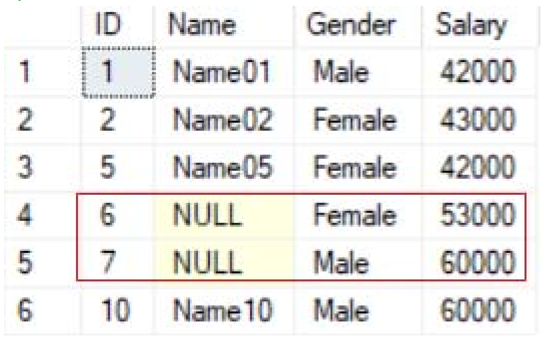
2.2. INTERSECT V.S. INNER JOIN V.S. DISTINCT INNER JOIN

```
--=================================================================
--T027_02_02
--INTERSECT  V.S. INNER JOIN  V.S.  DISTINCT INNER JOIN
```

2.2.1. INTERSECT

```
------------------------------------------------------------------
--T027_02_02_01
--INTERSECT
SELECT  ID ,
        [Name] ,
        Gender ,
        Salary
FROM    dbo.PersonA
INTERSECT
SELECT  ID ,
        [Name] ,
        Gender ,
        Salary
FROM    dbo.PersonB;
GO -- Run the previous command and begins new batch
/*
```

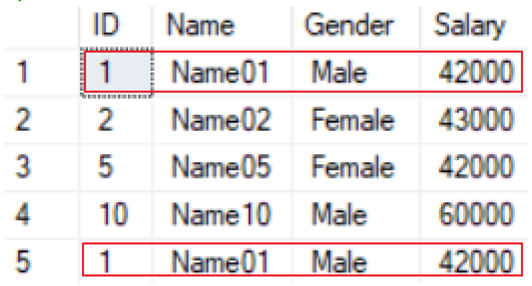
```
1.
1.1.
It will Show ID=1,2,5,6,7,10.
The [name] of row ID=6,7  is NULL.
1.2.
Then INTERSECT treats two NULLs as a same value
and it will think columnARow5 and columnBRow5 are matching row.
It will display this matching row.
*/
```



	ID	Name	Gender	Salary
1	1	Name01	Male	42000
2	2	Name02	Female	43000
3	5	Name05	Female	42000
4	6	NULL	Female	53000
5	7	NULL	Male	60000
6	10	Name10	Male	60000

2.2.2. INNER JOIN

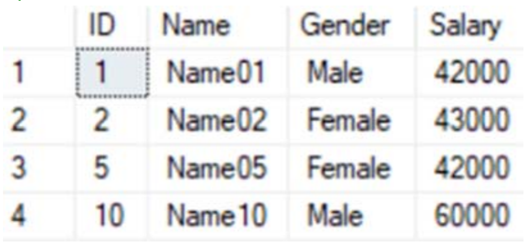
```
----------------------------------------------------------------------
--T027_02_02_02
--INNER JOIN
SELECT  dbo.PersonA.ID ,
        dbo.PersonA.[Name] ,
        dbo.PersonA.Gender ,
        dbo.PersonA.Salary
FROM    dbo.PersonA
        INNER JOIN dbo.PersonB ON dbo.PersonA.[Name] = dbo.PersonB.[Name];
GO -- Run the previous command and begins new batch
/*
1.
1.1.
It will Show ID=1,2,5,10,1.
The [name] of row ID=6,7  is NULL,
and these two rows does not display.
1.2.
(DISTINCT) INNER JOIN treats two NULLs as a different value
and it will think columnARow5 and columnBRow5 are NOT matching row.
It will NOT display this row.
1.3.
INTERSECT and DISTINCT INNER JOIN both return non-duplicated rows.
but INNER JOIN returns duplicated rows.
*/
```



	ID	Name	Gender	Salary
1	1	Name01	Male	42000
2	2	Name02	Female	43000
3	5	Name05	Female	42000
4	10	Name10	Male	60000
5	1	Name01	Male	42000

2.2.3. DISTINCT INNER JOIN

```
----------------------------------------------------------------------
--T027_02_02_03
--DISTINCT INNER JOIN
SELECT  DISTINCT
        dbo.PersonA.ID ,
        dbo.PersonA.[Name] ,
        dbo.PersonA.Gender ,
        dbo.PersonA.Salary
FROM    dbo.PersonA
        INNER JOIN dbo.PersonB ON dbo.PersonA.[Name] = dbo.PersonB.[Name];
GO -- Run the previous command and begins new batch
/*
1.
1.1.
It will Show ID=1,2,5,10.
The [name] of row ID=6,7  is NULL,
and these two rows does not display.
The ID=1 is a duplicated row,
and DISTINCT will only show one of them.
1.2.
(DISTINCT) INNER JOIN treats two NULLs as a different value
and it will think columnARow5 and columnBRow5 are NOT matching row.
It will NOT display this row.
1.3.
INTERSECT and DISTINCT INNER JOIN both return non-duplicated rows.
but INNER JOIN returns duplicated rows.
*/
```



	ID	Name	Gender	Salary
1	1	Name01	Male	42000
2	2	Name02	Female	43000
3	5	Name05	Female	42000
4	10	Name10	Male	60000

2.3. Clean up

```
--======================================================================
--T027_02_03
--Clean up
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'PersonA' ) )
    BEGIN
        TRUNCATE TABLE dbo.PersonA;
        DROP TABLE PersonA;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'PersonB' ) )
    BEGIN
        TRUNCATE TABLE dbo.PersonB;
        DROP TABLE PersonB;
    END;
GO -- Run the previous command and begins new batch
```

==

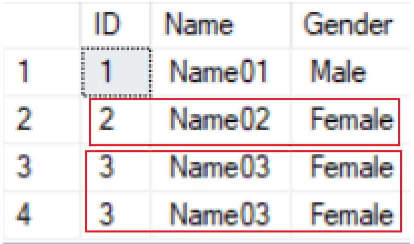
3. Intersect V.S. Except

```
--======================================================================
--T027_03_Intersect V.S. Except
--======================================================================
```

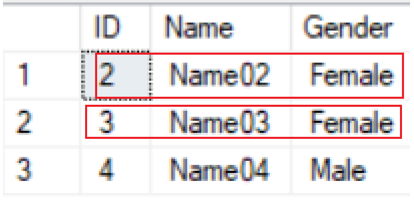
3.1. Create sample data

```
--=====================================================================
--T027_03_01
--Create sample data
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'PersonA' ) )
    BEGIN
        TRUNCATE TABLE dbo.PersonA;
        DROP TABLE PersonA;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'PersonB' ) )
    BEGIN
        TRUNCATE TABLE dbo.PersonB;
        DROP TABLE PersonB;
    END;
GO -- Run the previous command and begins new batch
CREATE TABLE PersonA
    (
      ID INT ,
      [Name] NVARCHAR(100) ,
      Gender NVARCHAR(10)
    );
GO -- Run the previous command and begins new batch
INSERT  INTO PersonA
VALUES  ( 1, 'Name01', 'Male' );
INSERT  INTO PersonA
VALUES  ( 2, 'Name02', 'Female' );
--Duplicated Rows
INSERT  INTO PersonA
VALUES  ( 3, 'Name03', 'Female' );
INSERT  INTO PersonA
VALUES  ( 3, 'Name03', 'Female' );
GO -- Run the previous command and begins new batch
CREATE TABLE PersonB
    (
      ID INT ,
      [Name] NVARCHAR(100) ,
      Gender NVARCHAR(10)
    );
GO -- Run the previous command and begins new batch
INSERT  INTO PersonB
VALUES  ( 2, 'Name02', 'Female' );
INSERT  INTO PersonB
VALUES  ( 3, 'Name03', 'Female' );
INSERT  INTO PersonB
VALUES  ( 4, 'Name04', 'Male' );
GO -- Run the previous command and begins new batch
SELECT  *
FROM    dbo.PersonA;
SELECT  *
FROM    dbo.PersonB;
```

```
GO -- Run the previous command and begins new batch
```



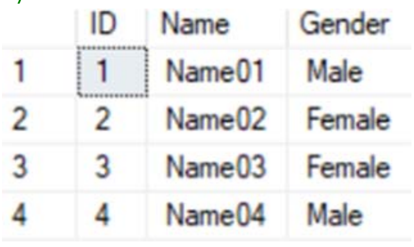
	ID	Name	Gender
1	1	Name01	Male
2	2	Name02	Female
3	3	Name03	Female
4	3	Name03	Female



	ID	Name	Gender
1	2	Name02	Female
2	3	Name03	Female
3	4	Name04	Male

3.2. UNION

```
--======================================================================
--T027_03_02
--UNION
SELECT  ID ,
        [Name] ,
        Gender
FROM    PersonA
UNION
SELECT  ID ,
        [Name] ,
        Gender
FROM    PersonB;
/*
1.
--UNION
UNION operator removes duplicates rows and
only returns unique rows from both PersonA and PersonB.
2.
Output
ID=1,2,3,4
*/
```

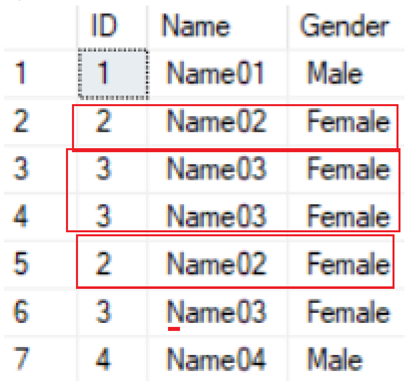


	ID	Name	Gender
1	1	Name01	Male
2	2	Name02	Female
3	3	Name03	Female
4	4	Name04	Male

3.3. UNION ALL

```
--======================================================================
--T027_03_03
--UNION ALL
SELECT  ID ,
        [Name] ,
        Gender
FROM    dbo.PersonA
UNION ALL
SELECT  ID ,
        [Name] ,
        Gender
FROM    dbo.PersonB;
/*
```

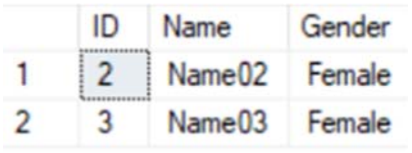
```
1.
--UNION ALL
UNION ALL operator does NOT remove duplicates rows and
returns all rows from both PersonA and PersonB.
2.
Output
ID=1,2,3,3,2,3,4
*/
```



	ID	Name	Gender
1	1	Name01	Male
2	2	Name02	Female
3	3	Name03	Female
4	3	Name03	Female
5	2	Name02	Female
6	3	Name03	Female
7	4	Name04	Male

3.4. INTERSECT

```
--=====================================================================
--T027_03_04
--INTERSECT
SELECT  ID ,
       Name ,
        Gender
FROM    dbo.PersonA
INTERSECT
SELECT  ID ,
       Name ,
        Gender
FROM    dbo.PersonB;
/*
--INTERSECT
1.
INTERSECT operator retrieves the rows
which exists in both PersonA and PersonB
and removes the duplicated rows.
2.
Output
ID=2,3
*/
```

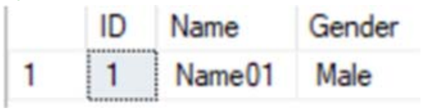


	ID	Name	Gender
1	2	Name02	Female
2	3	Name03	Female

3.5. EXCEPT

```
--=====================================================================
--T027_03_05
--EXCEPT
SELECT  ID ,
        [Name] ,
        Gender
FROM    dbo.PersonA
EXCEPT
SELECT  ID ,
        [Name] ,
        Gender
```

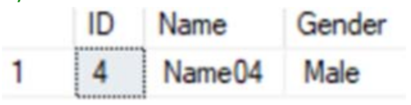
```
FROM    dbo.PersonB;
/*
1.
--EXCEPT
EXCEPT operator retrieves the unique rows from PersonA
that does not exist in PersonB
2.
Output
ID=1
*/
```



	ID	Name	Gender
1	1	Name01	Male

3.6. EXCEPT

```
--======================================================================
--T027_03_06
--EXCEPT
SELECT  ID ,
        [Name] ,
        Gender
FROM    dbo.PersonB
EXCEPT
SELECT  ID ,
        [Name] ,
        Gender
FROM    dbo.PersonA;
GO -- Run the previous command and begins new batch
/*
1.
--EXCEPT
EXCEPT operator retrieves the unique rows from PersonB
that does not exist in PersonA
2.
Output
ID=4
*/
```



	ID	Name	Gender
1	4	Name04	Male

3.7. Clean up

```
--======================================================================
--T027_03_07
--Clean up
--If Table exists then DROP it
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'PersonA' ) )
    BEGIN
        TRUNCATE TABLE dbo.PersonA;
        DROP TABLE PersonA;
    END;
GO -- Run the previous command and begins new batch
--If Table exists then DROP it
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'PersonB' ) )
    BEGIN
```

```
        TRUNCATE TABLE dbo.PersonB;
        DROP TABLE PersonB;
    END;
GO -- Run the previous command and begins new batch
```