(T23)合併 Merge
CourseGUID: e48417fc-9db5-4e99-822c-706c5ccef6cc
==========
(T23)合併 Merge
==========

==========

0. Summary

1.
```
--MERGE Person4_Target AS p4t
--USING Person4_Source AS p4s
--ON p4t.ID = p4s.ID
--WHEN MATCHED THEN
--    UPDATE SET p4t.Name = p4s.Name
--WHEN NOT MATCHED BY TARGET THEN
--    --When Source has, but Target has not.
--    --then insert into Target.
--    INSERT ( ID, Name )
--    VALUES ( p4s.ID, p4s.Name )
--WHEN NOT MATCHED BY SOURCE THEN
--    --When Source has not, but Target has.
--    --then delete it from the Target.
--    DELETE;
```
1.1.
Mirror Merge Syntax
```
--MERGE [targetTable] AS T
--USING [sourceTable] AS S
--   ON [JOIN_CONDITIONS]
-- WHEN MATCHED THEN
--      --[UPDATE STATEMENT: Update T by S ]
-- WHEN NOT MATCHED BY TARGET THEN
--      --[INSERT STATEMENT]
--      --insert rows to Target if rows do not exist in Target.
-- WHEN NOT MATCHED BY SOURCE THEN
--    --[DELETE STATEMENT] ;
--      --delete rows in Target if rows do not exist in Source.
```
Merge need ";"semicolumn to End the statement.
sourceTable Table is actuall a Changed Table which contain all the changes.
targetTable Table is a normal data storage.
When syncing, SourceTable will perform mirror merge into TargetTable.
Thus, TargetTable will become exactly the same as SourceTable.
1.1.1.
Delete the rows in TargetTable

if the rows do not exist in SourceTable,
but the rows exist in TargetTable.
1.1.2.
Insert rows to TargetTable
if the rows do not exist in TargetTable,
but the rows exist in SourceTable.
2.
--MERGE Person4_Target AS p4t
--USING Person4_Source AS p4s
--ON p4t.ID = p4s.ID
--WHEN MATCHED THEN
--   UPDATE SET p4t.Name = p4s.Name
--WHEN NOT MATCHED BY TARGET THEN
--   --When Source has, but Target has not.
--   --then insert into Target.
--   INSERT (ID, Name)
--   VALUES (p4s.ID, p4s.Name);
----WHEN NOT MATCHED BY SOURCE THEN
----   --When Source has not, but Target has.
----   --then delete it from the Target.
----   DELETE;
2.1.
Merge Syntax
--MERGE [targetTable] AS T
--USING [sourceTable] AS S
--   ON [JOIN_CONDITIONS]
-- WHEN MATCHED THEN
--   --[UPDATE STATEMENT: Update T by S]
-- WHEN NOT MATCHED BY TARGET THEN
--   --[INSERT STATEMENT] ;
--   --insert rows to Target if rows do not exist in Target.
Merge need ";"semicolumn to End the statement.
sourceTable Table is actuall a Changed Table which contain all the changes.
targetTable Table is a normal data storage.
When syncing, SourceTable will perform merge into TargetTable.
Thus, TargetTable might have more rows than its SourceTable.
2.1.1.
Do Nothing for the rows in TargetTable
if the rows do not exist in SourceTable,
but the rows exist in TargetTable.
2.1.2.
Insert rows to TargetTable
if the rows do not exist in TargetTable,
but the rows exist in SourceTable.

==========

1. Mirror Merge

```
--=============================================================================
--T023_01_Mirror Merge
--=============================================================================
/*
Perform "Mirror Merge" Source into Target
delete rows in Target if rows do not exist in Source.
insert rows to Target if rows do not exist in Target.
*/
```

1.1. Create Sample Data

```
--=============================================================================
--T023_01_01
--Create Sample Data
IF ( EXISTS ( SELECT    *
```

```
                FROM      INFORMATION_SCHEMA.TABLES
                WHERE     TABLE_NAME = 'Person4_Source' ) )
    BEGIN
              TRUNCATE TABLE Person4_Source
        DROP TABLE Person4_Source;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
                FROM      INFORMATION_SCHEMA.TABLES
                WHERE     TABLE_NAME = 'Person4_Target' ) )
    BEGIN
              TRUNCATE TABLE Person4_Target
        DROP TABLE Person4_Target;
    END;
GO -- Run the previous command and begins new batch
-------------------------------------
CREATE TABLE Person4_Source
    (
       ID INT PRIMARY KEY ,
       [Name] NVARCHAR(20)
    );
GO -- Run the previous command and begins new batch
INSERT  INTO Person4_Source
VALUES  ( 1, 'First1' );
INSERT  INTO Person4_Source
VALUES  ( 2, 'First2' );
INSERT  INTO Person4_Source
VALUES  ( 4, 'First4 Last4' );
INSERT  INTO Person4_Source
VALUES  ( 5, 'First5' );
GO -- Run the previous command and begins new batch
-------------------------------------
CREATE TABLE Person4_Target
    (
       ID INT PRIMARY KEY ,
      Name NVARCHAR(20)
    );
GO -- Run the previous command and begins new batch
INSERT  INTO Person4_Target
VALUES  ( 1, 'First1 Last1' );
INSERT  INTO Person4_Target
VALUES  ( 3, 'First3' );
INSERT  INTO Person4_Target
VALUES  ( 4, 'First4' );
GO -- Run the previous command and begins new batch
-------------------------------------
SELECT  *
FROM    dbo.Person4_Source;
SELECT  *
FROM    dbo.Person4_Target;
GO -- Run the previous command and begins new batch
```

	ID	Name
1	1	First1
2	2	First2
3	4	First4 Last4
4	5	First5

	ID	Name
1	1	First1 Last1
2	3	First3
3	4	First4

1.2. Mirror Merge

```
--=============================================================================
--T023_01_02
--Mirror Merge
/*
Perform "Mirror Merge" Source into Target
delete rows in Target if rows do not exist in Source.
insert rows to Target if rows do not exist in Target.
*/
MERGE Person4_Target AS p4t
USING Person4_Source AS p4s
ON p4t.ID = p4s.ID
WHEN MATCHED THEN
    UPDATE SET p4t.Name = p4s.Name
WHEN NOT MATCHED BY TARGET THEN
        --When Source has, but Target has not.
        --then insert into Target.
    INSERT ( ID, Name )
    VALUES ( p4s.ID, p4s.Name )
WHEN NOT MATCHED BY SOURCE THEN
        --When Source has not, but Target has.
        --then delete it from the Target.
    DELETE;
GO -- Run the previous command and begins new batch
SELECT  *
FROM    dbo.Person4_Source;
SELECT  *
FROM    dbo.Person4_Target;
GO -- Run the previous command and begins new batch
```

	ID	Name
1	1	First1
2	2	First2
3	4	First4 Last4
4	5	First5

	ID	Name
1	1	First1
2	2	First2
3	4	First4 Last4
4	5	First5

```
/*
1.
Merge Source into Target (mirror merge)
Person4_Source Table is actuall a Changed Table which contain all the changes.
Person4_Target Table is a normal data storage.
When syncing, we have to merge Person4_Source into Person4_Target.
delete rows in Target if rows do not exist in Source.
insert rows to Target if rows do not exist in Target.
Thus, Person4_Target will have the following values.
--VALUES  ( 1, 'First1' );
--VALUES  ( 2, 'First2' );
--VALUES  ( 4, 'First4 Last4' );
--VALUES  ( 5, 'First5' );
2.
Mirror Merge Syntax
--MERGE [targetTable] AS T
--USING [sourceTable] AS S
--   ON [JOIN_CONDITIONS]
-- WHEN MATCHED THEN
--       --[UPDATE STATEMENT: Update T by S ]
-- WHEN NOT MATCHED BY TARGET THEN
--      --[INSERT STATEMENT]
--       --insert rows to Target if rows do not exist in Target.
-- WHEN NOT MATCHED BY SOURCE THEN
--    --[DELETE STATEMENT] ;
--       --delete rows in Target if rows do not exist in Source.
Merge need ";"semicolumn to End the statement.
sourceTable Table is actuall a Changed Table which contain all the changes.
targetTable Table is a normal data storage.
When syncing, SourceTable will perform mirror merge into TargetTable.
Thus, TargetTable will become exactly the same as SourceTable.
2.1.
Delete the rows in TargetTable
if the rows do not exist in SourceTable,
but the rows exist in TargetTable.
2.2.
Insert rows to TargetTable
if the rows do not exist in TargetTable,
but the rows exist in SourceTable.
*/

==========
```

2. Normal Merge 2

```
--=============================================================================
--T023_02_Normal Merge2
--=============================================================================
/*
Perform "Merge" Source into Target
insert rows to Target if rows do not exist in Target.
Do NOT delete rows in Target if rows do not exist in Source.
*/
```

2.1. Create Sample Data

```
--=============================================================================
--T023_02_01
--Create Sample Data
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'Person4_Source' ) )
   BEGIN
            TRUNCATE TABLE Person4_Source
```

```
        DROP TABLE Person4_Source;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'Person4_Target' ) )
    BEGIN
            TRUNCATE TABLE Person4_Target
        DROP TABLE Person4_Target;
    END;
GO -- Run the previous command and begins new batch
-------------------------------------
CREATE TABLE Person4_Source
(
  ID INT PRIMARY KEY ,
  [Name] NVARCHAR(20)
);
GO -- Run the previous command and begins new batch
INSERT  INTO Person4_Source
VALUES  ( 1, 'First1' );
INSERT  INTO Person4_Source
VALUES  ( 2, 'First2' );
INSERT  INTO Person4_Source
VALUES  ( 4, 'First4 Last4' );
INSERT  INTO Person4_Source
VALUES  ( 5, 'First5' );
GO -- Run the previous command and begins new batch
-------------------------------------
CREATE TABLE Person4_Target
    (
       ID INT PRIMARY KEY ,
      Name NVARCHAR(20)
    );
GO -- Run the previous command and begins new batch
-------------------------------------
INSERT  INTO Person4_Target
VALUES  ( 1, 'First1 Last1' );
INSERT  INTO Person4_Target
VALUES  ( 3, 'First3' );
INSERT  INTO Person4_Target
VALUES  ( 4, 'First4' );
GO -- Run the previous command and begins new batch
-------------------------------------
SELECT  *
FROM    dbo.Person4_Source;
SELECT  *
FROM    dbo.Person4_Target;
GO -- Run the previous command and begins new batch
```

	ID	Name
1	1	First1
2	2	First2
3	4	First4 Last4
4	5	First5

	ID	Name
1	1	First1 Last1
2	3	First3
3	4	First4

2.2. Normal Merge 2

```
--=============================================================================
--T023_02_02
/*
Perform "Merge" Source into Target
insert rows to Target if rows do not exist in Target.
Do NOT delete rows in Target if rows do not exist in Source.
*/
MERGE Person4_Target AS p4t
USING Person4_Source AS p4s
ON p4t.ID = p4s.ID
WHEN MATCHED THEN
    UPDATE SET p4t.Name = p4s.Name
WHEN NOT MATCHED BY TARGET THEN
        --When Source has, but Target has not.
        --then insert into Target.
    INSERT ( ID, Name )
    VALUES ( p4s.ID, p4s.Name );
--WHEN NOT MATCHED BY SOURCE THEN
--      --When Source has not, but Target has.
--      --then delete it from the Target.
--    DELETE;
GO -- Run the previous command and begins new batch
---------------------------------------
SELECT  *
FROM    dbo.Person4_Source;
SELECT  *
FROM    dbo.Person4_Target;
GO -- Run the previous command and begins new batch
```

	ID	Name
1	1	First1
2	2	First2
3	4	First4 Last4
4	5	First5

	ID	Name
1	1	First1
2	2	First2
3	3	First3
4	4	First4 Last4
5	5	First5

```
/*
1.
Merge Source into Target
Person4_Source Table is actuall a Changed Table which contain all the changes.
Person4_Target Table is a normal data storage.
When syncing, we have to merge Person4_Source into Person4_Target.
insert rows to Target if rows do not exist in Target.
Thus, Person4_Target will have the following values.
--1    First1
--2    First2
--3    First3
--4    First4 Last4
--5    First5
2.
Merge Syntax
--MERGE [targetTable] AS T
--USING [sourceTable] AS S
--   ON [JOIN_CONDITIONS]
-- WHEN MATCHED THEN
--       --[UPDATE STATEMENT: Update T by S ]
-- WHEN NOT MATCHED BY TARGET THEN
--      --[INSERT STATEMENT] ;
--       --insert rows to Target if rows do not exist in Target.
Merge need ";"semicolumn to End the statement.
sourceTable Table is actuall a Changed Table which contain all the changes.
targetTable Table is a normal data storage.
When syncing, SourceTable will perform merge into TargetTable.
Thus, TargetTable might have more rows than its SourceTable.
2.1.
Do Nothing for the rows in TargetTable
if the rows do not exist in SourceTable,
but the rows exist in TargetTable.
2.2.
Insert rows to TargetTable
if the rows do not exist in TargetTable,
but the rows exist in SourceTable.
*/

==================================================
```

3. Clean up

```
--=============================================================================
--T023_03_Clean up
--=============================================================================
-- Drop Tables if it exists
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'Person4_Source' ) )
    BEGIN
            TRUNCATE TABLE Person4_Source
        DROP TABLE Person4_Source;
    END;
GO -- Run the previous command and begins new batch
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE     TABLE_NAME = 'Person4_Target' ) )
    BEGIN
            TRUNCATE TABLE Person4_Target
        DROP TABLE Person4_Target;
    END;
GO -- Run the previous command and begins new batch
```