(T3)討論IdentityColumn  
CourseGUID: e48417fc-9db5-4e99-822c-706c5ccef6cc  
=======================================================================  
(T3)討論IdentityColumn  
=======================================================================  
0. Introduction

0.1. What to learn

0.2. In Sumary

-----------

1. T003\_IdentityColumn\_01.sql

-----------

2. T003\_IdentityColumn\_02.sql  
=======================================================================

0. Introduction

0.1. What to learn

- Create Table with Identity Column

- TRUNCATE TABLE TableA;

- DBCC CHECKIDENT(TableA, RESEED, 1);

- SCOPE\_IDENTITY();

- @@IDENTITY;

- IDENT\_CURRENT('TableA');

- IDENT\_CURRENT('TableB');

0.2. In Sumary

1.

--TRUNCATE TABLE TableA;

--DBCC CHECKIDENT(TableA, RESEED, 1);

Truncate Table and DBcc CheckIdent should **always** be used together.

That means clear all data and reset Identity column.

2.

Don't use SCOPE\_IDENTITY()  or  @@IDENTITY

Because these are too confused.

Always use **IDENT\_CURRENT('TableName')**

This will always return the last identity value.

2.1.

In brief:

\* SCOPE\_IDENTITY()

returns the last identity value that is created in the same session and in the same scope.

\* @@IDENTITY

returns the last identity value that is created in the same session and across any scope.

@@IDENTITY stored the Identity Column Value from last affected scope in the same session.

\* IDENT\_CURRENT('tblPerson')

returns the last identity value that is created for a specific table across any session and any scope.

2.2.

In the same Seccion (Connection):

Every comand in current sql query edit window is

ONE CONNECTION to SQL server.

ONE CONNECTION means in the ONE session in this case.

2.3.

In the same Scope:

Same Scope means every sql command in ONE Stored procedure or ONE function, or ONE trigger.

1. T003\_IdentityColumn\_01.sql

--===================================================================================

--T003\_IdentityColumn

--===================================================================================

/\*

What to learn

- Create Table with Identity Column

- TRUNCATE TABLE TableA;

- DBCC CHECKIDENT(TableA, RESEED, 1);

\*/

--===================================================================================

--T003\_01

-- DBCC CHECKIDENT(TableA, RESEED, 1);

------------------------------------------------------------------------------------

--T003\_01\_01

-- Create Table with Identity Column

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'TableA' ) )

    BEGIN

        TRUNCATE TABLE TableA;

        DROP TABLE TableA;

    END;

GO -- Run the previous command and begins new batch

CREATE TABLE TableA

(

  Id INT IDENTITY(1, 1)

         PRIMARY KEY

         NOT NULL ,

  Value NVARCHAR(20)

)

    ON

[PRIMARY];

GO -- Run the prvious command and begins new batch

/\*

1.

--Id INT IDENTITY(1, 1)

--         PRIMARY KEY

--         NOT NULL ,

It means Id is the Primary Key and the type is int.

Id will start from 1 (the first one is identity seed),

and then increase 1 (the second one is identity increment)

2.

-- ON [PRIMARY]

When you create database, SQL server will generate

one .MDF(primary data file) and one .LDF(log file)

Sometimes a SQL Server database will include one or more .NDF (secondary data files).

-- ON [PRIMARY]

means create this table on the .MDF(primary data file).

\*/

------------------------------------------------------------------------------------

--T003\_01\_02

-- Insert data

INSERT  TableA

VALUES  ( N'Name01' );

SET IDENTITY\_INSERT TableA ON;

INSERT  TableA

        ( Id, [Value] )

VALUES  ( 2, N'Name02' );

SET IDENTITY\_INSERT TableA OFF;

SELECT  \*

FROM    TableA;

GO -- Run the prvious command and begins new batch

/\*

1.

You do not have to provide value for identity column

because it is auto generated.

If you want to provide value for identity column,

then you have to set IDENTITY\_INSERT is ON.

--SET IDENTITY\_INSERT TableA ON;

--INSERT ...

--SET IDENTITY\_INSERT TableA OFF;

2.

Now, we have ID=1 and ID=2 Record.

\*/

Table

Description automatically generated

------------------------------------------------------------------------------------

--T003\_01\_03

-- Delete a data and Insert data

DELETE  TableA

WHERE   Id = 2;

INSERT  TableA

VALUES  ( N'Name03' );

SELECT  \*

FROM    TableA;

GO -- Run the prvious command and begins new batch

/\*

1.

INT type Identity Column will not fill the gap

-->

When you delete the ID=2 record.

And the you insert another record,

It will auto generate ID=3.

Because ID=1 and ID=2 has been used.

The new record ID will not be 2.

2.

Now, we have ID=1 and ID=3 Record.

\*/

Table

Description automatically generated with low confidence

------------------------------------------------------------------------------------

--T003\_01\_04

-- DBCC CHECKIDENT(TableA, RESEED, 1);

SELECT  \*

FROM    TableA;

DBCC CHECKIDENT(TableA, RESEED, 1);

INSERT  TableA

VALUES  ( N'Name04' );

SELECT  \*

FROM    TableA;

GO -- Run the prvious command and begins new batch

/\*

1.

-- DBCC CHECKIDENT(TableA, RESEED, 1);

1.1.

Output message

--Checking identity information: current identity value '3'.

--DBCC execution completed. If DBCC printed error messages, contact your system administrator.

1.2.

You used to have ID=1 and ID=3 Record.

Then you execute

--DBCC CHECKIDENT(TableA, RESEED, 1);

Check the value of Identiy Column of Table 'TableA', then reset the seed to 1.

Seed=1 means the next ID of new record will be 2 (identity seed).

You have just insert ID=2 Record.

2.

Now,you have ID=1,ID=2,ID=3 Record.

and the Identiy Column Seed=2 at the moment

\*/

Table

Description automatically generated

------------------------------------------------------------------------------------

--T003\_01\_05

--Insert Data

SELECT  \*

FROM    TableA;

INSERT  TableA

VALUES  ( N'Name05' );

SELECT  \*

FROM    TableA;

/\*

1.

Output error message.

--Msg 2627, Level 14, State 1, Line 153

--Violation of PRIMARY KEY constraint 'PK\_\_TableA\_\_3214EC0703341784'.

--Cannot insert duplicate key in object 'dbo.TableA'.

--The duplicate key value is (3).

--The statement has been terminated.

You used to have ID=1,ID=2,ID=3 Record.

and the Identiy Column Seed=2 at the moment.

Identiy Column Seed=2 means your next insert will be ID=3.

Now, You are trying to insert next record.

However, ID=3 has already been there.

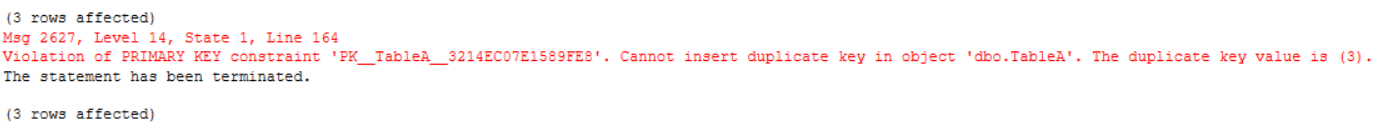
Thus, Error has occured.

2.

Now,you have ID=1,ID=2,ID=3 Record.

and the Identiy Column Seed=3 at the moment

\*/



------------------------------------------------------------------------------------

--T003\_01\_06

--Insert Data

SELECT  \*

FROM    TableA;

INSERT  TableA

VALUES  ( N'Name06' );

SELECT  \*

FROM    TableA;

/\*

1.

You used to have ID=1,ID=2,ID=3 Record.

and the Identiy Column Seed=3 at that moment.

Seed=3 means the next ID of new record will be 4 (identity seed).

ID=4 has NOT been there yet.

Thus, you can insert successfully.

2.

Now,you have ID=1,ID=2,ID=3,ID=4 Record.

and the Identiy Column Seed=4 at the moment.

Seed=4 means the next ID of new record will be 5 (identity seed).

\*/

Graphical user interface, table

Description automatically generated

------------------------------------------------------------------------------------

--T003\_01\_06

--- TRUNCATE TABLE TableA;

TRUNCATE TABLE TableA;

DBCC CHECKIDENT(TableA, RESEED, 1);

/\*

1.

-- TRUNCATE TABLE TableA;

and

--DELETE  TableA

are both doing the same thing to delete every data in the table.

However, TRUNCATE TABLE is better

because TRUNCATE TABLE will delete the data and clean up the space.

DELETE will delete the data without clean up the space.

It is more possible to cause data fragmentation.

2.

Output message

--Checking identity information: current identity value 'NULL'.

--DBCC execution completed. If DBCC printed error messages, contact your system administrator.

3.

--DBCC CHECKIDENT(TableA, RESEED, 1);

Check the value of Identiy Column of Table 'TableA', then reset the seed to 1.

Seed=1 means the next ID of new record will be 2 (identity seed).

However,

You have executed  TRUNCATE Table (delete every data and clean up space).

This will make your current identity value 'NULL'

Only when current identity value 'NULL',

then  Seed=1 means the next ID of new record will be 1 (identity seed).

That means

DBCC CHECKIDENT need to be used very carefully.

You better execute  TRUNCATE Table (delete every data and clean up space).

Then execute

--DBCC CHECKIDENT(TableA, RESEED, 1);

\*/



------------------------------------------------------------------------------------

--T003\_01\_07

--Insert

SELECT  \*

FROM    TableA;

INSERT  TableA

VALUES  ( N'Name07' );

SELECT  \*

FROM    TableA;

/\*

1.

You do not have to provide value for identity column

because it is auto generated.

2.

Now, we have ID=1 Record.

\*/

Graphical user interface, application, table

Description automatically generated

--===================================================================================

--T003\_02

/\*

What to learn

- SCOPE\_IDENTITY();

- @@IDENTITY;

- IDENT\_CURRENT('TableA');

- IDENT\_CURRENT('TableB');

\*/

----------------------------------------------------------------------------------

--T003\_02\_01

--Create Sample Data

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'TableA' ) )

    BEGIN

        TRUNCATE TABLE TableA;

        DROP TABLE TableA;

    END;

GO -- Run the previous command and begins new batch

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'TableB' ) )

    BEGIN

        TRUNCATE TABLE TableB;

        DROP TABLE TableB;

    END;

GO -- Run the previous command and begins new batch

CREATE TABLE TableA

(

  ID INT IDENTITY(1, 1)

         PRIMARY KEY

         NOT NULL ,

  [Value] NVARCHAR(20)

);

CREATE TABLE TableB

(

  ID INT IDENTITY(1, 1)

         PRIMARY KEY

         NOT NULL ,

  [Value] NVARCHAR(20)

);

----------------------------------------------------------------------------------

--T003\_02\_02

SELECT  SCOPE\_IDENTITY();

--NULL

SELECT  @@IDENTITY;

--NULL

SELECT  IDENT\_CURRENT('TableA');

--1

SELECT  IDENT\_CURRENT('TableB');

--1

GO -- Run the prvious command and begins new batch

/\*

1.

[ID] [int] IDENTITY(1,1) NOT NULL,

It means ID is the Primary Key and the type is int.

ID will start from 1 (the first one is identity seed),

and then increase 1 (the second one is identity increment)

2.

2.1.

In brief:

\* SCOPE\_IDENTITY()

returns the last identity value that is created in the same session and in the same scope.

\* @@IDENTITY

returns the last identity value that is created in the same session and across any scope.

@@IDENTITY stored the Identity Column Value from last affected scope in the same session.

\* IDENT\_CURRENT('tblPerson')

returns the last identity value that is created for a specific table across any session and any scope.

2.2.

In the same Seccion (Connection):

Every comand in current sql query edit window is

ONE CONNECTION to SQL server.

ONE CONNECTION means in the ONE session in this case.

2.3.

In the same Scope:

Same Scope means every sql command in ONE Stored procedure or ONE function, or ONE trigger.

3.

Originally,

Table A have nothing

Table B have nothing.

Therefore,

SCOPE\_IDENTITY()  return 0

@@IDENTITY  return 0

IDENT\_CURRENT('TableA') return 1

IDENT\_CURRENT('TableB')  return 1

\*/

----------------------------------------------------------------------------------

--T003\_02\_02

INSERT  INTO TableA

VALUES  ( 'x1' );

SELECT  \*

FROM    TableA;

SELECT  SCOPE\_IDENTITY();

--1

SELECT  @@IDENTITY;

--1

SELECT  IDENT\_CURRENT('TableA');

--1

SELECT  IDENT\_CURRENT('TableB');

--1

GO -- Run the prvious command and begins new batch

/\*

1.

Table A have (ID=1,Value='x1')

Table B have nothing.

Thus,

SCOPE\_IDENTITY() return 1

@@IDENTITY return 1

IDENT\_CURRENT('TableA') return 1

IDENT\_CURRENT('TableB') return 1

\*/

----------------------------------------------------------------------------------

--T003\_02\_03

INSERT  INTO TableA

VALUES  ( 'x2' );

SELECT  \*

FROM    TableA;

SELECT  SCOPE\_IDENTITY();

--2

SELECT  @@IDENTITY;

--2

SELECT  IDENT\_CURRENT('TableA');

--2

SELECT  IDENT\_CURRENT('TableB');

--1

GO -- Run the prvious command and begins new batch

/\*

1.

Table A have (ID=1,Value='x1'), (ID=2,Value='x2')

Table B have nothing.

Thus,

SCOPE\_IDENTITY() return 2

@@IDENTITY return 2

IDENT\_CURRENT('TableA') return 2

IDENT\_CURRENT('TableB') return 1

\*/

----------------------------------------------------------------------------------

--T003\_02\_04

IF EXISTS ( SELECT  \*

            FROM    sys.objects

            WHERE   [name] = N'tgForInsert'

                    AND [type] = 'TR' )

    BEGIN

        DROP TRIGGER tgForInsert;

    END;

GO -- Run the previous command and begins new batch

CREATE TRIGGER tgForInsert ON TableA

    AFTER INSERT

AS

    BEGIN

        INSERT  INTO TableB

                SELECT  [Value]

                FROM    Inserted;

    END;

GO -- Run the prvious command and begins new batch

/\*

1.

Create a trigger which

will insert the same value to TableB

when you insert to TableA.

\*/

----------------------------------------------------------------------------------

--T003\_02\_05

SELECT  \*

FROM    TableA;

SELECT  \*

FROM    TableB;

--Table A have (ID=1,Value='x1'), (ID=2,Value='x2')

--Table B have nothing.

SELECT  SCOPE\_IDENTITY();

--2

SELECT  @@IDENTITY;

--2

SELECT  IDENT\_CURRENT('TableA');

--2

SELECT  IDENT\_CURRENT('TableB');

--1

INSERT  INTO TableA

VALUES  ( 'x3' );

SELECT  \*

FROM    TableA;

SELECT  \*

FROM    TableB;

--Table A have (ID=1,Value='x1'), (ID=2,Value='x2'), (ID=3,Value='x3')

--Table B have (ID=3,Value='x3').

SELECT  SCOPE\_IDENTITY();

--3

SELECT  @@IDENTITY;

--1

SELECT  IDENT\_CURRENT('TableA');

--3

SELECT  IDENT\_CURRENT('TableB');

--1

/\*

1.

1.1.

In brief:

\* SCOPE\_IDENTITY()

returns the last identity value that is created in the same session and in the same scope.

\* @@IDENTITY

returns the last identity value that is created in the same session and across any scope.

@@IDENTITY stored the Identity Column Value from last affected scope in the same session.

\* IDENT\_CURRENT('tblPerson')

returns the last identity value that is created for a specific table across any session and any scope.

1.2.

In the same Seccion (Connection):

Every comand in current sql query edit window is

ONE CONNECTION to SQL server.

ONE CONNECTION means in the ONE session in this case.

1.3.

In the same Scope:

Same Scope means every sql command in ONE Stored procedure or ONE function, or ONE trigger.

2.

Originally,

We have insert 2 values into Table A.

Thus,

Table A have (ID=1,Value='x1'), (ID=2,Value='x2')

Table B have nothing.

SCOPE\_IDENTITY()  return 2

@@IDENTITY  return 2

IDENT\_CURRENT('TableA') return 2

IDENT\_CURRENT('TableB')  return 1

Then

We create a TableA Trigger "tgForInsert" which

will insert the same value to TableB

when you insert to TableA.

3.

After insert (ID=3,Value='x3') into TableA

Table A will have (ID=1,Value='x1'), (ID=2,Value='x2'), (ID=3,Value='x3')

Table B will have (ID=1,Value='x3') by the TableA Trigger "tgForInsert"

Then,

SCOPE\_IDENTITY()  return 3

--@@IDENTITY  return 1

--IDENT\_CURRENT('TableA') return 3

--IDENT\_CURRENT('TableB')  return 1

3.1.

SCOPE\_IDENTITY()

returns the last identity value that is created in the same session and in the same scope.

--SCOPE\_IDENTITY()  return 3

because this number is from that

We have inserted 3 times to TableA

in the same session and in the same scope.

3.2.

--@@IDENTITY  return 1

because this number is from that

the TableA Trigger "tgForInsert" was triggered

and insert  (ID=1,Value='x3') to TableB.

@@IDENTITY stored the Identity Column Value from last affected scope in the same session.

Same Scope means every sql command in ONE Stored procedure or ONE function, or ONE trigger.

Same session(CONNECTION) means every sql command in ONE sql query edit window.

3.3.

IDENT\_CURRENT('tblPerson')

returns the last identity value that is created for a specific table across any session and any scope.

--IDENT\_CURRENT('TableA') return 3

--IDENT\_CURRENT('TableB')  return 1

Both return the last identity value

across any session and any scope.

This is the safest way to get the last identity column value.

\*/

2. T003\_IdentityColumn\_02.sql

-- T003\_IdentityColumn -----------------------------------------------------

--===================================================================================

--T003\_03

/\*

What to learn

- SCOPE\_IDENTITY();

- @@IDENTITY;

- IDENT\_CURRENT('TableA');

- IDENT\_CURRENT('TableB');

\*/

----------------------------------------------------------------------------------

--T003\_03\_01

-- Ch08\_02\_01

SELECT  SCOPE\_IDENTITY();

--NULL

SELECT  @@IDENTITY;

--NULL

SELECT  IDENT\_CURRENT('TableA');

--3

SELECT  IDENT\_CURRENT('TableB');

--1

GO -- Run the prvious command and begins new batch

/\*

1.

1.1.

In brief:

\* SCOPE\_IDENTITY()

returns the last identity value that is created in the same session and in the same scope.

\* @@IDENTITY

returns the last identity value that is created in the same session and across any scope.

@@IDENTITY stored the Identity Column Value from last affected scope in the same session.

\* IDENT\_CURRENT('tblPerson')

returns the last identity value that is created for a specific table across any session and any scope.

1.2.

In the same Seccion (Connection):

Every comand in current sql query edit window is

ONE CONNECTION to SQL server.

ONE CONNECTION means in the ONE session in this case.

1.3.

In the same Scope:

Same Scope means every sql command in ONE Stored procedure or ONE function, or ONE trigger.

2.

SCOPE\_IDENTITY() return NULL

@@IDENTITY return NULL

IDENT\_CURRENT('TableA') return 3

IDENT\_CURRENT('TableB') return 1

\*/

----------------------------------------------------------------------------------

--T003\_03\_02

SELECT  SCOPE\_IDENTITY();

--NULL

SELECT  @@IDENTITY;

--NULL

SELECT  IDENT\_CURRENT('TableA');

--3

SELECT  IDENT\_CURRENT('TableB');

--1

SELECT  \*

FROM    TableA;

SELECT  \*

FROM    TableB;

--Table A have (ID=1,Value='x1'), (ID=2,Value='x2'), (ID=3,Value='x3')

--Table B have (ID=3,Value='x3').

INSERT  INTO TableB

VALUES  ( 'x2' );

SELECT  SCOPE\_IDENTITY();

--2

SELECT  @@IDENTITY;

--2

SELECT  IDENT\_CURRENT('TableA');

--3

SELECT  IDENT\_CURRENT('TableB');

--2

SELECT  \*

FROM    TableA;

SELECT  \*

FROM    TableB;

--Table A have (ID=1,Value='x1'), (ID=2,Value='x2'), (ID=3,Value='x3')

--Table B have (ID=3,Value='x3'), (ID=2,Value='x2')

GO -- Run the prvious command and begins new batch

/\*

1.

Originally,

Table A will have (ID=1,Value='x1'), (ID=2,Value='x2'), (ID=3,Value='x3')

Table B will have (ID=1,Value='x3')

SCOPE\_IDENTITY()  return NULL

@@IDENTITY  return NULL

IDENT\_CURRENT('TableA') return 3

IDENT\_CURRENT('TableB')  return 1

2.

After insert (ID=2,Value='x2') into TableB

Table A will have (ID=1,Value='x1'), (ID=2,Value='x2'), (ID=3,Value='x3')

Table B will have (ID=1,Value='x3'), (ID=2,Value='x2')

SCOPE\_IDENTITY()  return 2

@@IDENTITY  return 2

IDENT\_CURRENT('TableA') return 3

IDENT\_CURRENT('TableB')  return 2

2.1.

SCOPE\_IDENTITY()  return 2

because this number is from that

We have inserted 2 times to TableB

in the same session and in the same scope.

2.2.

@@IDENTITY  return 2

because this number is from that

We have inserted 2 times to TableB

in the same session and in the same scope.

@@IDENTITY stored the Identity Column Value from last affected scope in the same session.

Same Scope means every sql command in ONE Stored procedure or ONE function, or ONE trigger.

Same session(CONNECTION) means every sql command in ONE sql query edit window.

2.3.

IDENT\_CURRENT('TableA') return 3

IDENT\_CURRENT('TableB')  return 2

Both return the last identity value

across any session and any scope.

This is the safest way to get the last identity column value.

\*/

----------------------------------------------------------------------------------

--T003\_03\_03

SELECT  SCOPE\_IDENTITY();

--2

SELECT  @@IDENTITY;

--2

SELECT  IDENT\_CURRENT('TableA');

--3

SELECT  IDENT\_CURRENT('TableB');

--2

SELECT  \*

FROM    TableA;

SELECT  \*

FROM    TableB;

--Table A have (ID=1,Value='x1'), (ID=2,Value='x2'), (ID=3,Value='x3')

--Table B have (ID=3,Value='x3'), (ID=2,Value='x2')

INSERT  INTO TableA

VALUES  ( 'x4' );

SELECT  SCOPE\_IDENTITY();

--4

SELECT  @@IDENTITY;

--3

SELECT  IDENT\_CURRENT('TableA');

--4

SELECT  IDENT\_CURRENT('TableB');

--3

SELECT  \*

FROM    TableA;

SELECT  \*

FROM    TableB;

--Table A have (ID=1,Value='x1'), (ID=2,Value='x2'), (ID=3,Value='x3'), (ID=4,Value='x4')

--Table B have (ID=3,Value='x3'), (ID=2,Value='x2'), (ID=3,Value='x4')

/\*

1.

Originally,

--Table A have (ID=1,Value='x1'), (ID=2,Value='x2'), (ID=3,Value='x3')

--Table B have (ID=3,Value='x3'), (ID=2,Value='x2')

SCOPE\_IDENTITY()  return 2

@@IDENTITY  return 2

IDENT\_CURRENT('TableA') return 3

IDENT\_CURRENT('TableB')  return 2

2.

After We insert (ID=4,Value='x4') into TableA

The TableA Trigger "tgForInsert" will insert (ID=3,Value='x4') into TableB

Thus,

Table A will have (ID=1,Value='x1'), (ID=2,Value='x2'), (ID=3,Value='x3'), (ID=4,Value='x4')

Table B will have (ID=1,Value='x3'), (ID=2,Value='x2'), (ID=3,Value='x4')

SCOPE\_IDENTITY()  return 4

@@IDENTITY  return 3

IDENT\_CURRENT('TableA') return 4

IDENT\_CURRENT('TableB')  return 3

2.1.

SCOPE\_IDENTITY()  return 4

because this number is from that

We have inserted 4 times to TableA

in the same session and in the same scope.

2.2.

@@IDENTITY  return 3

because this number is from that

We have inserted 3 times to TableB

in the same session and in the same scope.

@@IDENTITY stored the Identity Column Value from last affected scope in the same session.

Same Scope means every sql command in ONE Stored procedure or ONE function, or ONE trigger.

Same session(CONNECTION) means every sql command in ONE sql query edit window.

2.3.

IDENT\_CURRENT('TableA') return 4

IDENT\_CURRENT('TableB')  return 3

Both return the last identity value

across any session and any scope.

This is the safest way to get the last identity column value.

\*/

--===================================================================================

--T003\_04

--Clean up

IF EXISTS ( SELECT  \*

            FROM    sys.objects

            WHERE   [name] = N'tgForInsert'

                    AND [type] = 'TR' )

    BEGIN

        DROP TRIGGER tgForInsert;

    END;

GO -- Run the previous command and begins new batch

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'TableA' ) )

    BEGIN

        TRUNCATE TABLE TableA;

        DROP TABLE TableA;

    END;

GO -- Run the previous command and begins new batch

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'TableB' ) )

    BEGIN

        TRUNCATE TABLE TableB;

        DROP TABLE TableB;

    END;

GO -- Run the previous command and begins new batch