IDENTITY (Property)

Source **http://msdn2.microsoft.com/en-us/library/ms186775.aspx**

IDENTITY (Property) (Transact-SQL)

Creates an identity column in a table. This property is used with the CREATE TABLE and ALTER TABLE Transact-SQL statements.

Note: The IDENTITY property is different from the SQL-DMO Identity property that exposes the row identity property of a column.

Syntax

```
IDENTITY [ (seed , increment) ]
```

Arguments

seed

Is the value that is used for the very first row loaded into the table.

increment

Is the incremental value that is added to the identity value of the previous row that was loaded.

You must specify both the seed and increment or neither. If neither is specified, the default is (1,1).

Remarks

If an identity column exists for a table with frequent deletions, gaps can occur between identity values. If this is a concern, do not use the IDENTITY property. However, to make sure that no gaps have been created or to fill an existing gap, evaluate the existing identity values before explicitly entering one with SET IDENTITY_INSERT ON.

If you are reusing a removed identity value, use the sample code in Example B to look for the next available identity value. Replace tablename, column_type, and MAX(column_type) - 1 with a table name, identity column data type, and numeric value of the maximum allowed value (for that data type) -1.

Use DBCC CHECKIDENT to check the current identity value and compare it with the maximum value in the identity column.

If a table with an identity column is published for replication, the identity column must be managed in a way that is appropriate for the type of replication used.

Examples

A. Using the IDENTITY property with CREATE TABLE

The following example creates a new table using the IDENTITY property for an automatically incrementing identification number.

```
USE AdventureWorks

IF OBJECT_ID ('dbo.new_employees', 'U') IS NOT NULL

   DROP TABLE new_employees

GO

CREATE TABLE new_employees

(

 id_num int IDENTITY(1,1),

 fname varchar (20),

 minit char(1),

 lname varchar(30)

)

INSERT new_employees

   (fname, minit, lname)

VALUES

   ('Karin', 'F', 'Josephs')

INSERT new_employees

   (fname, minit, lname)

VALUES

   ('Pirkko', 'O', 'Koskitalo')
```

Using generic syntax for finding gaps in identity values

The following example shows generic syntax for finding gaps in identity values when data is removed.

Note: The first part of the following Transact-SQL script is designed for illustration only. You can run the Transact-SQL script that starts with the comment: -- Create the img table.

```
-- Here is the generic syntax for finding identity value gaps in data.

-- The illustrative example starts here.

SET IDENTITY_INSERT tablename ON

DECLARE @minidentval column_type

DECLARE @maxidentval column_type

DECLARE @nextidentval column_type

SELECT @minidentval = MIN($IDENTITY), @maxidentval = MAX($IDENTITY)

    FROM tablename

IF @minidentval = IDENT_SEED('tablename')

   SELECT @nextidentval = MIN($IDENTITY) + IDENT_INCR('tablename')

   FROM tablename t1

   WHERE $IDENTITY BETWEEN IDENT_SEED('tablename') AND

      @maxidentval AND

      NOT EXISTS (SELECT * FROM tablename t2

         WHERE t2.$IDENTITY = t1.$IDENTITY +

            IDENT_INCR('tablename'))

ELSE

   SELECT @nextidentval = IDENT_SEED('tablename')

SET IDENTITY_INSERT tablename OFF

-- Here is an example to find gaps in the actual data.

-- The table is called img and has two columns: the first column

-- called id_num, which is an increasing identification number, and
the

-- second column called company_name.

-- This is the end of the illustration example.
-- Create the img table.

-- If the img table already exists, drop it.

-- Create the img table.

IF OBJECT_ID ('dbo.img', 'U') IS NOT NULL

   DROP TABLE img

GO

CREATE TABLE img (id_num int IDENTITY(1,1), company_name sysname)

INSERT img(company_name) VALUES ('New Moon Books')

INSERT img(company_name) VALUES ('Lucerne Publishing')

-- SET IDENTITY_INSERT ON and use in img table.

SET IDENTITY_INSERT img ON

DECLARE @minidentval smallint

DECLARE @nextidentval smallint

SELECT @minidentval = MIN($IDENTITY) FROM img

 IF @minidentval = IDENT_SEED('img')

    SELECT @nextidentval = MIN($IDENTITY) + IDENT_INCR('img')

    FROM img t1

    WHERE $IDENTITY BETWEEN IDENT_SEED('img') AND 32766 AND

      NOT EXISTS (SELECT * FROM img t2

          WHERE t2.$IDENTITY = t1.$IDENTITY + IDENT_INCR('img'))

 ELSE

    SELECT @nextidentval = IDENT_SEED('img')

SET IDENTITY_INSERT img OFF
```

***~~~ End of Article ~~~***