DateCreated column in Sql Server?

6

3

Is there a special way to declare a DateCreated column in a MS Sql Server table so that it will automatically fill it with the appropriate time-stamp when created?

Or.. do I have to provide the datetime to it when I do the query, manually?

Default values suffer from two major drawbacks.

- if the insert statement specifies a value for the column, the default isn't used.
- the column can be updated any time.

These mean that you can't be certain that the values haven't been modified outside of your control.

If you want true data integrity (so that you're sure the date in the row is the creation date), you need to use triggers.

An insert trigger to set the column to the current date and an update trigger to prevent changes to that column (or, more precisely, set it to its current value) are the way to implement a DateCreated column.

An insert and update trigger to set the column to the current date is the way to implement a DateModified column.

(edit from user Gabriel - here's my attempt to implement this as described - i'm not 100% sure it's correct but I'm hoping the OP reviews it...):

```
CREATE TRIGGER [dbo].[tr_Affiliate_IU]
   ON  [dbo].[Affiliate]
   AFTER INSERT, UPDATE
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Get the current date.
    DECLARE @getDate DATETIME = GETDATE()
    -- Set the initial values of date_created and
date_modified.
    UPDATE
        dbo.Affiliate
    SET
         date_created = @getDate
    FROM
        dbo.Affiliate A
        INNER JOIN INSERTED I ON A.id = I.id
        LEFT OUTER JOIN DELETED D ON I.id = D.id
    WHERE
        D.id IS NULL

    -- Ensure the value of date_created does never changes.
    -- Update the value of date_modified to the current date.
    UPDATE
        dbo.Affiliate
    SET
         date_created = D.date_created
        ,date_modified = @getDate
    FROM
        dbo.Affiliate A
        INNER JOIN INSERTED I ON A.id = I.id
        INNER JOIN DELETED D ON I.id = D.id
END
```

We have DEFAULT on CreatedDate and don't enforce with Triggers

There are times when we want to set the date explicitly - e.g. if we import data from some other source.

There is a risk that Application Bug could mess with the CreateDate, or a disgruntled DBA for that matter (we don't have non-DBAs connecting direct to our DBs)

I suppose you might set Column-level permissions on CreateDate.

A half-way-house might be to have an INSERT TRIGGER create a row in a 1:1 table, so that column was outside the main table. The second table could have SELECT permissions, where the main table has UPDATE permissions, and thus not need an UPDATE trigger to prevent changes to CreateDate - which would remove some "weight" when updating rows normally.

I suppose you coul have an UPDATE/DELETE trigger on the second table to prevent change (which would never be executed in normal circumstances, so "lightweight")

Bit of a pain to have the extra table though ... could have one table for all CreateDates - TableName, PK, CreateDate. Most database architects will hate that though ...