

Re: Calculated fiels in a table

Source:

<http://www.tech-archive.net/Archive/Access/microsoft.public.access.tablesdbdesign/2007-08/msg00016.html>

- *From:* "Dale Fye" <dale.fye@xxxxxxxxxxx>
 - *Date:* Wed, 1 Aug 2007 08:21:46 -0400
-

Jamie,

OK, triggers are reactive, DUHHHHHHH! If I change Col1 and Col3 depends on Col1, I should update Col3. This is preferable to a check constraint, that makes no changes in the value of Col3, but generates an error that prevents you from changing Col1 or Col2. I can see where this might be fine in an unbound form, where you write all the values to their appropriate fields in a single insert or update statement, but in a bound form, when I change the value of Col1, it attempts to change that value in the table (which would violate the check constraint) and therefore should generate an error.

I get an error (Invalid SQL Syntax – cannot use multiple columns in a column-level check constraint) within Access 2003. How would you implement this validation rule within Access, since we are talking about Access in this group, not SQL SERVER.

Dale

"Jamie Collins" <jamiecollins@xxxxxxxxxxx> wrote in message
<news:1185924302.534752.284600@xx>

On Jul 31, 10:26 pm, Dale Fye <dale....@xxxxxxxxxxx> wrote:

Here's a scenario (and I recommend AGAINST IT!):

- * You add a calculated field to an Access table, based on two other fields
- * Someone fixes one of the other fields (data entry error, sorry!)
- * Your calculated field is now out of synch, unless you create a routine that runs every time after any of the component parts change, AND after a 'manual' change in the value of the

Re: Calculated fiels in a table

calculated field.

In summary, you seem to be saying that one shouldn't store calculated values in a SQL DBMS but if you do you should do it in such a way that the values don't get out of sync, in which case I concur. With Access/Jet, such a "routine" could be an engine level Validation Rule or CHECK constraint.

I could see where, in a SQL Server database, where you can create triggers to maintain data integrity, you might want to do this, but with Access, there are just too many ways around the data entry process to ensure that when one of the fields used in your calculated field gets updated that it doesn't corrupt your data.

I don't think that is correct. If an engine-level constraint could be circumvented by any means (short of dropping it) then it would be a bug in the engine; I know of no such bug in the Access/Jet engine. I've heard of duplicated values in an autonumber PRIMARY KEY column due to file corruption but I would wager serious money that the PRIMARY KEY had been lost in the process. I know of CHECK constraints being checked too early but that could be considered erring on the side of caution <g>.

Take a very simple example:

```
CREATE TABLE MyCalcTable
(
col1 INTEGER NOT NULL,
col2 INTEGER NOT NULL,
col3 INTEGER NOT NULL,
CHECK (col3 = col1 + col2)
);
```

(One could use a Validation Rule in place of that CHECK constraint.) Can you suggest any "ways around the data entry process" that would result in col3 being anything other than the sum of col1 and col2? I cannot.

FWIW (slightly OT) I don't see anything special about implementing a constraint as a trigger when compared with a CHECK constraint. In SQL

Re: Calculated fiels in a table

Server, the optimizer can take account of CHECK constraints but has no knowledge of triggers so they can be slow as a result. Triggers are reactive and procedural, constraints are proactive and declarative. In a trigger you have to explicitly raise errors to the callee, rollback transactions, etc whereas with a constraint all this *housekeeping* is done for me by the DBMS. Most of the AFTER triggers I write (INSTEAD OF triggers are a different animal e.g. most often used to update an otherwise un-updatable VIEW) I wouldn't need if SQL Server supported full SQL-92 as regards CHECK constraints -- i.e. allow subqueries (as Access/Jet does) and be deferrable (as Oracle does) -- and ASSERTIONS.

Jamie.

--