SQL Server triggers - order of execution

Asked 16 years, 3 months ago Modified 4 years, 4 months ago

Viewed 42k times

35

Does anyone know how SQL Server determines the order triggers (of same type, i.e. before triggers) are executed. And is there any way of changing this so that I can specify the order I want. If not, why not.

Thanks.

sql sql-server sql-server-2005 t-sql

Share

Improve this question

Follow

edited Dec 23, 2015 at 8:50

Jim Aho

11.8k • 19 • 60 • 95

asked Sep 18, 2008 at 16:21

HAdes

17k • 23 • 62 • 75

10 Answers

Sorted by: Highest score (default)

Using SetTriggerOrder is fine, but if your code depends on a specific sequence of execution, why not wrap all

19

your triggers into stored procedures, and have the first one call the second, the second call the third, etc.

Then you simply have the first one execute in the trigger.

Someone in the future will be grateful that they didn't have to dig around in a system table to determine a custom execution sequence.

Share Improve this answer Follow

edited Sep 18, 2008 at 18:09

answered Sep 18, 2008 at 16:56

JosephStyons

58.6k ● 64 ● 167 ● 237

10 Generally we don't do this because we don't have easy access to the Inserted and Deleted pseudotables in stored procedures. – mwigdahl Jun 27, 2012 at 13:57

@mwigdahl Instead of writing the Insert/Update/Delete Statement ad-hoc, you should implement a design pattern where you run all your Business-Logic DML inside of Sprocs. There you will have access to the values you want to Insert/Update and can easily query for the data you will be Deleting. I'm not a fan of ORM because (as a DB purist), I think all DML should be encapsulated inside of Sprocs (except for fixes). If you can't do that (maybe it's a 3rd Party DB or you're the new guy no one will listen to about investing time in rearchitecture), then welcome to trigger-H-E-double-hockysticks. – MikeTeeVee Aug 2, 2018 at 20:38

16

You can use sp_settriggerorder to define the order of each trigger on a table.

However, I would argue that you'd be much better off having a single trigger that does multiple things. This is *particularly* so if the order is important, since that importance will not be very obvious if you have multiple triggers. Imagine someone trying to support the database months/years down the track. Of course there are likely to be cases where you need to have multiple triggers or it really is better design, but I'd start assuming you should have one and work from there.

Share Improve this answer

Follow

answered Sep 18, 2008 at 18:04

Rory

41.8k ● 55 ● 186 ● 269

While it's a good idea to have a single trigger if order of execution matters, this is not the case if it doesn't. Ex. - I have a project with two triggers on a table. One logs changes in case someone doesn't use the provided SP to update the table. The other acts as a FK between 3 tables with a complex relationship that cannot be described via "in-built" FK-s. The SP that updates the table disables the log trigger while it works (to avoid duplicate logs), but I still want the "key trigger" to run. If order of execution doesn't matter, group triggers by task to improve "code readability". – sisisisi Feb 26, 2021 at 5:56

If you're at the point of worrying about trigger orders then you really should take a step backwards and consider

13

what you are trying to do and if there is there a better way of doing it. The fact that this isn't an easy thing to change should be telling you something.

Triggers always look like a really neat solution, and in the right place they are highly valuable, **but** the price is high, and it's really easy to create debugging nightmares with them. I've lost many hours in the past trying to debug some obscure database behavior only to find that the cause is burrowed away in an overlooked trigger.

Share Improve this answer

Follow

edited Aug 18, 2020 at 21:33

RoastBeast

1,115 ● 2 ● 24 ● 41

answered Sep 18, 2008 at 16:49

Cruachan

16k ● 6 ● 62 ● 112

old post, new comment - I'd like to set the order so that if something goes wrong I know what triggers have run to reach the failure point. The fact that it's not easy to change tells me that Microsoft allow multiple triggers but didn't implement it terribly well. Taking a step backwards isn't always an option. Is it any better in SQL Server 2019

– Paul McCarthy Feb 17, 2023 at 9:24

sp_settriggerorder only applies to AFTER triggers.

7

Share Improve this answer

Follow

answered Sep 18, 2008 at 16:33

Luke Bennett

32.9k ● 3 ● 32 ● 57

4

You can guarantee which trigger is fired first, which trigger is fired last and which ones fire in the middle by using sp_settriggerorder. If you need to synchronize more than three it does not appear possible in SQL Server 2005.

Here is a sample taken from here (The linked article has much more information).

```
sp_settriggerorder [ @triggername = ] '[ triggerschema
, [ @order = ] 'value'
, [ @stmttype = ] 'statement_type'
[ , [ @namespace = ] { 'DATABASE' | 'SERVER' | NULL }
```

Share Improve this answer

Follow

answered Sep 18, 2008 at 16:34

Frosty

6,443 ● 3 ● 26 ● 20

4

Use `sp_Settriggerorder` stored procedure, you can define the execution order of the trigger.

```
sp_settriggerorder [ @triggername = ] '[ triggerschema
, [ @order = ] 'value'
, [ @stmttype = ] 'statement_type'
[ , [ @namespace = ] { 'DATABASE' | 'SERVER' | NULL }
```

The second parameter, “order” can take three values which means that it can take into account up-to three triggers.

1. First – Trigger is fired first
2. Last - Trigger is fired last
3. None – Trigger is fired in random order.

Share Improve this answer

Follow

edited Nov 25, 2015 at 12:02

Sabyasachi Mishra

1,749 ● 2 ● 32 ● 51

answered Sep 18, 2008 at 16:28

EvilEddie

1,035 ● 2 ● 13 ● 24

3

The order is set by sql server, the only thing you can do is use a system sp (sp_settriggerorder) to set which trigger will fire first and which will fire last.

Beyond setting the first and last triggers to fire, you can't modify or tell which order sql server will use. Therefore you will want to build your triggers so they do not rely on which order they are fired. Even if you determine the order they fire in today, it may change tomorrow.

This information is based on Sql Server 2000, however I do not believe 2005/2008 act differently in this regard.

Share Improve this answer

answered Sep 18, 2008 at 16:28

Follow

Dave_H
6,713 ● 1 ● 18 ● 6

2

Use This :

For example :

```
USE AdventureWorks;
GO
EXEC sys.sp_settriggerorder @triggername = N'', -- nva
    @order = '', -- varchar(10)
    @stmttype = '', -- varchar(50)
    @namespace = '' -- varchar(10)
```

The First and Last triggers must be two different triggers.

First : Trigger is fired first.

Last : Trigger is fired last.

None : Trigger is fired in undefined order.

And see this link for value of `@stmttype` : DDL Events

And for @namespace = { 'DATABASE' | 'SERVER' | NULL } and for more information see : DDL Triggers

Share Improve this answer

Follow

answered Sep 6, 2013 at 12:50

Ardalan Shahgholi
12.5k ● 24 ● 114 ● 152

Use this system stored procedure:

```
sp_settriggerorder[@triggername = ] 'triggername', [@o
[@stmttype = ] 'statement_type'
```

Share Improve this answer

Follow

answered Sep 18, 2008 at 16:28

Ben Hoffstein

103k ● 8 ● 106 ● 121

A million dollar statement in this context -

> sp_settriggerorder: Specifies the AFTER triggers that are fired first or last. The AFTER triggers that are fired between the first and last triggers are executed in undefined order.

Source : MSDN

Share Improve this answer

Follow

answered Apr 6, 2013 at 23:31

Akshay

50 ● 4