Job Clear Down Utility Creates Orphaned Subjobs In Database With KTA Upgrade

3024364

Question / Problem:

After running the Kofax TotalAgility (KTA) Job Clear Down utility in the System Settings module, it is discovered that subjobs are not deleted from the database (Finished_Jobs table), but the parent jobs are deleted, which creates orphaned subjobs that cannot be deleted even if the utility is run again.

Answer / Solution:

If KTA is upgraded from a version below v7.6 to a 7.6 or higher version, there is some missing code in the SQL Conversion script, which causes the subjobs to remain when the parent jobs are deleted. The script is used in the dbo.AWSP_REMOVECOMPLETEDJOBS2 stored procedure, which is called by the Job Clear Down utility.

Note: This issue does not occur with a clean installation of KTA 7.6. A fresh installation includes the missing script code. This only occurs with systems that have upgraded to KTA v7.6 or higher from a lower version.

The following bug report has been submitted to address this issue.

Bug 1547750: TotalAgility 7.6 Stored Procedure Missing Subjob Deletion Script

To correct this behavior, the stored procedure must be modified using the following steps. This will allow subjobs to be deleted for new jobs, but not existing subjobs. The orphaned subjobs can be removed with the script show further below.

Note: It is highly recommended to first create a full backup of the Archive database as well as test the modified procedure in a Test/DEV environment prior to implementing in a Production environment.

To update the current AWSP_REMOVECOMPLETEDJOBS2 stored procedure please follow these steps.

- Take a full backup of the KTA Archive database
- Right click on the AWSP_REMOVECOMPLETEDJOBS2 stored procedure and choose "Modify"
- After the following section in the script, insert the missing script section shown below.

--SELECT the first record to delete
SET @jobIdToDelete = (SELECT TOP 1 JOB_ID FROM @job);

-- Get the subjobs for current job

INSERT @subjobs

SELECT DISTINCT JOB_ID FROM FINISHED_JOB WITH (NOLOCK) WHERE ROOT_JOB_ID = @jobIdToDelete
AND TYPE = 3

• Execute the modified script to save the changes.
• Test deleting new finished jobs using the Job Clear Down utility.

The following SQL script can be used for deleting existing orphaned subjobs from the Finished_Jobs table. This should first be tested in a Test/DEV environment and a full backup of the Archive database should be taken prior to executing.

--First grab the list of job_ids to delete

DECLARE @job TABLE (JOB_ID BINARY(16))

--delete all orphaned subjob data

INSERT @job

SELECT DISTINCT JOB_ID FROM FINISHED_JOB WITH (NOLOCK) WHERE ROOT_JOB_ID NOT IN (SELECT
JOB_ID FROM FINISHED_JOB NOLOCK) AND TYPE = 3

-- Declare the total rows count.

DECLARE @count INT;

SELECT @count = COUNT(*) from @job;

--DECLARE the variable for jobid to delete.

DECLARE @jobIdToDelete BINARY(16);

-- Loop through the rows and delete each unique jobid as a single transaction.
WHILE @count > 0

BEGIN

BEGIN TRANSACTION

--SELECT the first record to delete
SET @jobIdToDelete = (SELECT TOP 1 JOB_ID FROM @job);

--Now delete the job data
DELETE FROM FINISHED_JOB_VARIABLE WHERE OWNER_ID = @jobIdToDelete
DELETE FROM FINISHED_JOB WHERE JOB_ID = @jobIdToDelete
DELETE FROM FINISHED_JOB_MILESTONES WHERE JOB_ID = @jobIdToDelete
DELETE FROM FINISHED_JOB_EVENT WHERE JOB_ID = @jobIdToDelete

-- DELETE the record from the @job table
DELETE FROM @job WHERE JOB_ID = @jobIdToDelete

-- DECREMENT THE COUNT BY 1
SET @count = @count - 1;

COMMIT TRANSACTION

END

GO
Applies to:

<table>
<thead>
<tr>
<th>Product</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>KTA</td>
<td>7.6</td>
</tr>
<tr>
<td></td>
<td>7.7</td>
</tr>
<tr>
<td></td>
<td>7.8</td>
</tr>
</tbody>
</table>