The following info regarding “ghost Record Cleanup” .

**From:** Nikesh Mhatre [<mailto:Nikesh.Mhatre@microsoft.com>]   
**Sent:** Thursday, October 18, 2012 2:01 PM  
**To:** Mahajan, Prashant: IT (LDN)  
**Cc:** MSSolve Case Email  
**Subject:** [112100855544376]- Summary email

Dear Prashant,

It was my pleasure to work with you during your “**Databases not freeing space after data deletion for text columns.**" issue. I hope the services provided to you have met or exceeded your expectation; and we appreciate any of your feedback about how to further improve our service quality. I am providing you with a summary of the key points of the case for your records. If you ever have any questions please feel free to call me.

Based on your email yesterday, I understand that this case is ready for closure. If this is premature or if you need additional assistance with this case, please let me know as soon as possible.

**[ISSUE]**

- You have a Clustered SQL Server 2008 (10.0.2740) named LDNPCM05899v05b\fi\_farm2\_live.  
- This has a database emfi\_TCWGmtEnricherPDN, which has a table called JOBS that contains BLOB data.  
- Even when you delete huge number of records the space is not claimed back to OS

**[RESOLUTION]**

Based on our discussion, I had sent below description of the issues and action plan to resolve it.

This issue may happen because of the ghost record cleanup not working properly.

More about ghost record cleanup:

The main reason behind introducing the concept of Ghost records was to enhance performance. In the leaf level of an index, when rows are deleted, they're marked as ghost records. This means that the row stays on the page but a bit is changed in the row header to indicate that the row is really a ghost. The page header also reflects the number of ghost records on a page. What this means, in effect, is that the DML operation which fired the delete will return to the user much faster, because it does not have to wait for the records to be deleted physically. Rather, they’re just marked as “ghosted”.

Ghost records are present only in the index leaf nodes. If ghost records weren't used, the entire range surrounding a deleted key would have to be locked. Here’s an example i picked up from somewhere:

The ghost records could be cleaned up in 3 ways:

• If a record of the same key value as the deleted record is inserted

• If the page needs to be split, the ghost records will be handled

• The Ghost cleanup task (scheduled to run once every 5 seconds)

So sometime when you delete a huge records from the SQL Server database and you see that it is not freeing up space quickly or at the rate it should have been doing, then you want to check if there are ghost records in db. Generally I follow the below steps

1. Run the following command:

Select \* from sys.dm\_db\_index\_physical\_stats(db\_id(<dbname>),<ObjectID>,NULL,NULL,’DETAILED’)

2. Check the Ghost\_Record\_Count and Version\_Ghost\_Record\_Count columns (version ghost record count will be populated when you’re using snapshot isolation on the database). If this is high (several million in some cases), then you’ve most probably got a ghost record cleanup issue. If this is SQL Server 2008/2008 R2, then make sure you have applied the patch mentioned in the kb <http://support.microsoft.com/kb/2622823>.  We did check that the Ghost\_Record\_count keeps on increasing.

3) At this moment we are running the below query to check if it helps. This may take a while to complete.

EXEC sp\_clean\_db\_free\_space @dbname=N’<dbname>’

4) You could also run below query as  a workaround.

COMMANDS:

--------------

DBCC TRACEON(661, 662,3604,-1)

go

DBCC FORCEGHOSTCLEANUP('<db\_name>')

go

DBCC TRACEOFF(661, 662,3604,-1)

Once the commands are completed, run the rebuild Index to see if the space is reclaimed.

Also please check the ghost record entries in the following dmv: select  getdate(),\* from sys.dm\_db\_index\_physical\_statsn (*databaseid*,object\_id(*'objectname*'),null,null,'detailed')

5) In case you are seeing the Ghost Records even after following the above action plan then I suspect, there is some mismatch between PFS Page ghost\_record Entries count for those records.

To identify that corruption, please run the attached stored procedure(dbo.usp\_getPFSGhostPages) for database and table in question and provide the output.

=================================================================

--This procedure prints all pages and corresponding PFS page information for a given input database and table.

--Usage : exec usp\_getPFSGhostPages @dbname, @tablename

--Example:

--exec usp\_getPFSGhostPages 'pubs', 'authors'

SET NOCOUNT ON

GO

CREATE PROCEDURE dbo.usp\_getPFSGhostPages @dbname varchar(100), @tablename varchar(100)

AS

BEGIN

      SET NOCOUNT ON

      SET ANSI\_WARNINGS OFF

     --Declare variables

      declare @file\_id varchar(10), @page\_id varchar(50), @dbccpagecmd varchar(100), @dbccextentcmd varchar(100)

      --Create Temp Tables

      create table #extentinfo (file\_id int, page\_id int, pg\_alloc  int,ext\_size int, object\_id varchar(15),

      index\_id int, partition\_number int, partition\_id varchar(25),iam\_chain\_type varchar(15),pfs\_bytes varbinary)

      create table #PageInfo (ParentObject varchar(25), Object varchar(50), Field varchar(50), Value varchar(50))

      create table #PFSPage (ObjectID varchar(20), PageID varchar(20), SlotCount int, GhostRecordCount int, PFSPageID varchar(50))

      --Populate extentinfo for input database and table

set @dbccextentcmd = 'DBCC EXTENTINFO('+ @dbname + ','+ Quotename (@tablename,'''') + ',-1) with no\_infomsgs'

      insert into #extentinfo exec (@dbccextentcmd)

      --Run DBCC PAGE for each page in #extentinfo. Retrieve necessary fields  into #PFSPage.

      --At the end of the cursor execution, #PFSPage should have (objectid, pageid, slotcount, ghostrecordcount, pfspageid) for every page of the input table.

      declare pageloop cursor for

      select file\_id, page\_id from #extentinfo

      open pageloop

      fetch next from pageloop into @file\_id, @page\_id

      WHILE @@FETCH\_STATUS = 0

      BEGIN

            --select @file\_id, @page\_id

            set @dbccpagecmd = 'DBCC PAGE('+ @dbname + ','+ @file\_id + ','+@page\_id+',0) with no\_infomsgs, tableresults'

            --select @dbccpagecmd

            insert #PageInfo exec (@dbccpagecmd)

            insert #PFSPage

            select max(case when Field = 'Metadata: ObjectId' then Value end) as ObjectID,

                   max(case when Field = 'm\_PageId' then Value end) as PageID,

                   max(case when Field = 'm\_slotCnt' then Value end) as SlotCount,

                   max(case when Field = 'm\_ghostRecCnt' then Value end) as GhostRecordCount,

                   max(case when Field like 'PFS%' then substring(Field, (CHARINDEX('(', Field)+1), LEN(Field)-(CHARINDEX('(', Field)+1)) end) as PFSPageID

            from #PageInfo

            where Field in ('Metadata: ObjectId','m\_pageId','m\_slotCnt', 'm\_ghostRecCnt') or Field like 'PFS%'

            TRUNCATE table #PageInfo

            fetch next from pageloop into @file\_id, @page\_id

      END

      CLOSE pageloop

      DEALLOCATE pageloop

      --Print all input table's pages with ghost records and their PFS pages

      PRINT 'DATA PAGE and PFS Page information for ' + @tablename

      SELECT PageID, SlotCount, GhostRecordCount, PFSPageID FROM #PFSPage

      WHERE GhostRecordCount > 0

      --Turn on TF 3604 for DBCC PAGE output.

      DBCC TRACEON(3604)

      declare pageloop cursor for

      select DISTINCT PFSPageID from #PFSPage where GhostRecordCount > 0

      open pageloop

      fetch next from pageloop into @page\_id

      WHILE @@FETCH\_STATUS = 0

      BEGIN

            PRINT 'Executing DBCC PAGE on PFS Page ' + @page\_id

            set @dbccpagecmd = 'DBCC PAGE('+ @dbname + ','+ REPLACE(@page\_id, ':', ',')+',3) with no\_infomsgs'

            --select @dbccpagecmd

            exec(@dbccpagecmd)

            fetch next from pageloop into @page\_id

      END

      CLOSE pageloop

      DEALLOCATE pageloop

      DBCC TRACEOFF(3604)

      --Drop Temp Tables

      drop table #extentinfo

      drop table #PageInfo

      drop table #PFSPage

END

* Then later run usp\_getPFSGhostPages 'databasename', 'dbo.tablename'

=================================================================

The long term solution would be to install the fix <http://support.microsoft.com/kb/2622823>.

**Conclusion:** Upgrading to this build would take some time as we need to wait for approvals. Meanwhile, DBCC FORCEGHOSTCLEANUP helps you to free up space.

Thanks for your time and patience..

Enjoy your day!!

Cheers

cid:image001.jpg@01CDAD5E.ABE8BC60

**Nikesh Mhatre**

SQL Server Engineer | Microsoft EMEA Global Technical Support Centre

 Email: [nimhatre@microsoft.com](mailto:nimhatre@microsoft.com)  Working Hours: Mon - Fri 8:00 A.M to 5:00 P.M GMT

 Direct:(44) 207 365 2380 x 2866

Manager: Sri Krishna Jagannath  [srijagan@microsoft.com](mailto:srijagan@microsoft.com)

Note: This email may contain confidential information. If you are not named on the addressee list, please take no action in relation to this email, do not open any attachment and please contact the sender (details above) immediately.