<https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/>

Oops… I forgot to leave an empty SQL table partition, how can I split it with minimal IO impact?

★★★★★

★★★★

★★★

★★

★

avatar of sqlpfeoutlook-com[SQL Server Premier Field Engineering](https://social.msdn.microsoft.com/profile/SQL+Server+Premier+Field+Engineering)

[SQL Server Premier Field Engi...](https://social.msdn.microsoft.com/profile/SQL+Server+Premier+Field+Engineering)

[Microsoft](https://social.msdn.microsoft.com/profile/SQL+Server+Premier+Field+Engineering)

[}](https://social.msdn.microsoft.com/profile/SQL+Server+Premier+Field+Engineering)

[MSFT](https://social.msdn.microsoft.com/profile/SQL+Server+Premier+Field+Engineering)

[7,024 Points 3 2 2](https://social.msdn.microsoft.com/profile/SQL+Server+Premier+Field+Engineering)

[Recent Achievements](https://social.msdn.microsoft.com/profile/SQL+Server+Premier+Field+Engineering)

[Blog Party Starter Blogger III Blog Conversation Starter](https://social.msdn.microsoft.com/profile/SQL+Server+Premier+Field+Engineering)

[View Profile](https://social.msdn.microsoft.com/profile/sql%20server%20premier%20field%20engineering/?ws=usercard-hover)

August 13, 2013[9](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/#comments)

[0](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/)

[0](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/)

One of the several advantages you get with partitioning a very large table is the ability to add or remove partitions instantaneously to help you with sliding window scenarios. However there are caveats if best practices aren’t followed. If the right most partition (in case of a RANGE LEFT partition) or the left most partition (in case of a RANGE RIGHT partition) is not empty, then adding a partition is time and resource intensive Since it will be a logged operation and there will be data movement

The Problem:

Given I did not follow the best practices and don’t have an empty partition, the Split operation on a non-empty partition is expensive in particular if we have partitions with millions or billions of rows. How would I execute a split in a way that is non-logged and incurs minimal overhead?

For this scenario, the table is partitioned by a date time column with RANGE LEFT but the same holds true with other data types as well partitioned with RANGE LEFT.

Note: We recommend range right partitions for Datetime values. For additional details as to why see the whitepaper below. However for the purpose of the blog we are using RANGE LEFT given this problem occurs only on the split of the right most non-empty partition which is much more common.

<http://technet.microsoft.com/en-US/library/ms345146(v=SQL.90).aspx#sql2k5parti_topic17>

Partition Function:

CREATE PARTITION FUNCTION [Orders\_\_Function](datetime) AS RANGE LEFT FOR VALUES

(N’2012-12-31T23:59:59.997′,

N’2013-03-31T23:59:59.997′,

N’2013-06-30T23:59:59.997′)

And here is how the data is partitioned in a table that uses the above partition function.

Partitioned Table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Partition # | Partition 1 | Partition 2 | Partition 3 | Partition 4 |
| Data Range | <= 12/31/2012 | <=3/31/2013 | <=6/30/2013 | > 6/30/2013 |

In order to understand the implications, let’s create a partitioned table and demonstrate how a split can result in data movement.

Note: The table we create has just 1000 rows in each partition, benefit is for much larger tables though.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Step 1: Scenario SETUP

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

use master

go

drop database PartitionTest

create database PartitionTest

use PartitionTest

go

— Add Filegroups

ALTER DATABASE [PartitionTest] ADD FILEGROUP [FG1];

ALTER DATABASE [PartitionTest] ADD FILEGROUP [FG2];

ALTER DATABASE [PartitionTest] ADD FILEGROUP [FG3];

ALTER DATABASE [PartitionTest] ADD FILEGROUP [FG4];

GO

Alter database PartitionTest set recovery simple

go

— Add Files

ALTER DATABASE [PartitionTest] ADD FILE ( NAME = N’PartitionTest\_1′, FILENAME = N’D:\Temp\PartitionTest\_1.ndf’) TO FILEGROUP [FG1]

ALTER DATABASE [PartitionTest] ADD FILE ( NAME = N’PartitionTest\_2′, FILENAME = N’D:\Temp\PartitionTest\_2.ndf’) TO FILEGROUP [FG2]

ALTER DATABASE [PartitionTest] ADD FILE ( NAME = N’PartitionTest\_3′, FILENAME = N’D:\Temp\PartitionTest\_3.ndf’) TO FILEGROUP [FG3]

ALTER DATABASE [PartitionTest] ADD FILE ( NAME = N’PartitionTest\_4′, FILENAME = N’D:\Temp\PartitionTest\_4.ndf’) TO FILEGROUP [FG4]

GO

— Create partition function

CREATE PARTITION FUNCTION [Orders\_\_Function](datetime) AS RANGE LEFT FOR VALUES

(N’2012-12-31T23:59:59.997′,

N’2013-03-31T23:59:59.997′,

N’2013-06-30T23:59:59.997′)

go

— Create partition Scheme

CREATE PARTITION SCHEME [Orders\_\_Scheme] AS PARTITION [Orders\_\_Function] TO

([FG1],[FG2],[FG3],[FG4])

— Create table

CREATE TABLE [dbo].[Orders](

[OrdDate] [datetime] NOT NULL,

[ID] [bigint] IDENTITY(1,1) NOT NULL,

[Addr] varchar(100) NOT NULL)

— Partition the table

CREATE UNIQUE CLUSTERED INDEX IX\_Orders

ON [Orders](OrdDate asc,ID asc)

ON [Orders\_\_Scheme] (OrdDate);

GO

— Insert rows into partitions (partition 4 in this case)

Use PartitionTest

set nocount on

go

declare @i int

set @i = 1

declare @date Datetime

while (@i < 1000)

begin

set @date = dateadd(mi,@i,‘2012-11-01T10:17:01.000’)

–insert into testtable values (@date)

insert into [Orders] values (@date, ‘Denzil’)

insert into [Orders] values (dateadd(month,3,@date), ‘Denzil’)

insert into [Orders] values (dateadd(month,6,@date), ‘Denzil’)

insert into [Orders] values (dateadd(month,9,@date), ‘Denzil’)

set @i = @i+1;

end

— Check the rowcount in each partition

select $PARTITION.[Orders\_\_Function](Orddate) as PartionNum,COUNT(\*) as CountRows from Orders

Group by $PARTITION.[Orders\_\_Function](Orddate)

Now if we look at the distribution of rows, we will see that the last partition is not empty and has 999 rows

— View Metadata before split

SELECT

t.name as TableName,i.name as IndexName,

p.partition\_id as partitionID,

p.partition\_number,rows, fg.name

FROM sys.tables AS t

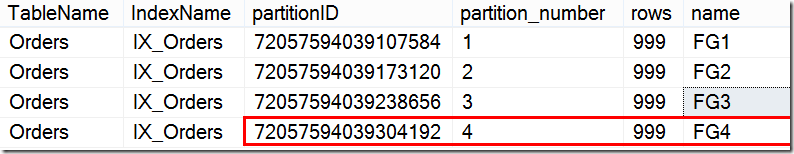
INNER JOIN sys.indexes AS i ON (t.object\_id = i.object\_id)

INNER JOIN sys.partitions AS p ON (t.object\_id = p.object\_id and i.index\_id = p.index\_id)

INNER JOIN sys.destination\_data\_spaces dds ON (p.partition\_number = dds.destination\_id)

INNER JOIN sys.filegroups AS fg ON (dds.data\_space\_id = fg.data\_space\_id)

WHERE (t.name = ‘Orders’) and (i.index\_id IN (0,1))

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/24/03/metablogapi/0083.clip_image001_11407C03.png)

Now executing a Normal split, you can notice that for the 999 rows in that partition split, each record is deleted and then inserted into the new partition.

— We now want to SPLIT a non-empty partition, so preparing for that

— Add new Filegroup and file

ALTER DATABASE [PartitionTest] ADD FILEGROUP [FG5];

ALTER DATABASE [PartitionTest] ADD FILE

( NAME = N’PartitionTest\_5′, FILENAME = N’D:\Temp\PartitionTest\_5.ndf’ , SIZE = 3072KB , FILEGROWTH = 1024KB )

TO FILEGROUP [FG5]

GO

— Set the next used partition

Alter partition scheme [Orders\_\_Scheme] NEXT USED [FG5]

— Traditional Split

— Clear Records to demonstrate Log records generated

checkpoint

go

— Select to demonstrate that there are no log records for that table

select Operation,count(\*) as NumLogRecords from fn\_dblog(NULL,NULL)

where AllocUnitName= ‘dbo.Orders.IX\_Orders’

group by Operation

order by count(\*) desc

[clip_image003](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/24/03/metablogapi/0844.clip_image003_1477854D.jpg)

— Split the non-empty partition

ALTER PARTITION FUNCTION Orders\_\_Function() SPLIT RANGE (‘2013-09-30 23:59:59.99’)

The result of the query below that shows us the Logs records indicate that each row was deleted and then inserted into the newly created partition.

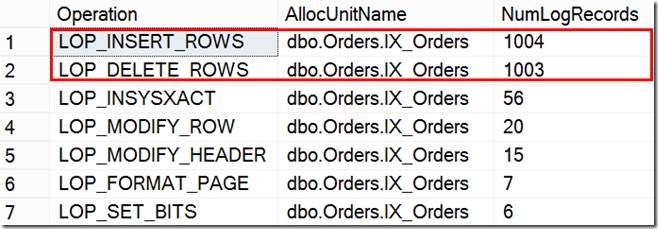
— Show how many log records generated, there is data movement Deletes followed by inserts

select Operation,AllocUnitName,count(\*) as NumLogRecords from fn\_dblog(NULL,NULL)

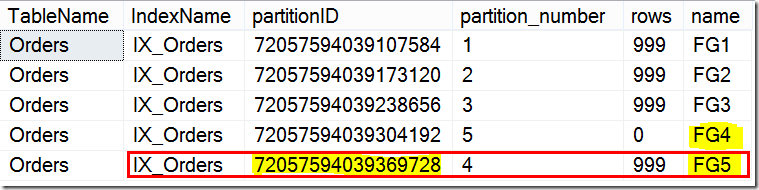
where AllocUnitName= ‘dbo.Orders.IX\_Orders’

group by Operation,AllocUnitName

order by count(\*) desc

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/24/03/metablogapi/8715.clip_image005_5FD9B9C2.jpg)

Also looking at the Metadata, you can see that the partition number 4 has been now assigned a new partition\_id whereas the newly added partition (partition number 5) was assigned the partition\_id that was previously assigned to partition number 4

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/24/03/metablogapi/5611.clip_image006_2EDA0915.png)

The Solution:

So, the solution is to switch the data from Partition 4 into an empty table to make the last partition empty. You can then split the partition to add new boundary and then switch the data back into Partition 4.

In order to accomplish this efficiently, create a temporary staging table in the same file group (FG\_unlimited in the example) as the last partition. Switch out the last partition (partition\_number 4 in the example) into this staging table by using. The switch operation will finish in few seconds as it is only a metadata operation.

— Create a copy table with intent to switch in

CREATE TABLE [dbo].[Orders\_Copy](

[OrdDate] [datetime] NOT NULL,

[ID] [bigint] IDENTITY(1,1) NOT NULL,

[Addr] varchar(100) NOT NULL)

— Create the clustered index on the copy table on the same filegroup as the partition that we are trying to switch out.

CREATE UNIQUE CLUSTERED INDEX IX\_Orders\_Copy

ON [Orders\_COPY](OrdDate asc,ID asc)

ON [FG4];

GO

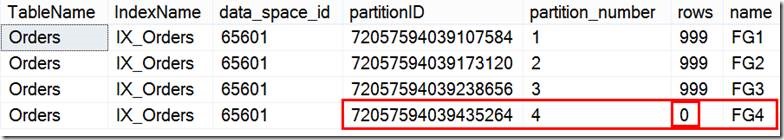
— Execute the switch. After this, the last partition should be empty.

ALTER TABLE Orders SWITCH PARTITION 4 TO Orders\_Copy;

— All the data in partition 4 is now gone to the table Orders\_Copy

select count(\*) as NumRows from Orders\_copy

[clip_image008](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/24/03/metablogapi/0842.clip_image008_3E7CA4D7.jpg)

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/24/03/metablogapi/7635.clip_image010_27BD5D4E.jpg)

Now we split an empty partition below, and there are no log record generated and this is a metadata operation only and if you notice the last 2 partitions have 0 rows.

— Mark the Filegroup used by the last partition as the NEXT USED. This is for the SWITCH to work.

Alter partition scheme [Orders\_\_Scheme]

NEXT USED [FG4]

— Clear TLog Records

checkpoint

go

— Split the now partition

ALTER PARTITION FUNCTION Orders\_\_Function() SPLIT RANGE (‘2013-09-30 23:59:59.997’)

— Will see no Logged data movement

select Operation,AllocUnitName,count(\*) as NumLogRecords from fn\_dblog(NULL,NULL)

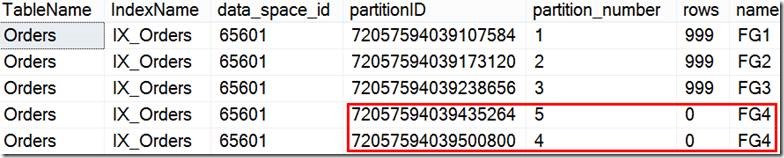
where AllocUnitName= ‘dbo.Orders.IX\_Orders’

group by Operation,AllocUnitName

order by count(\*) desc

[clip_image012](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/24/03/metablogapi/3157.clip_image012_7076D612.png)

And viewing the metadata:

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/24/03/metablogapi/0486.clip_image014_232D8159.jpg)

You can now switch the data from the staging table back into partition 4.

Note: You need to ensure that you have a CHECK constraint created on the staging table with the appropriate date range, before you can switch the data back into the partition. Please check the following link for all requirements for Partition Switching operations.

<http://msdn.microsoft.com/en-us/library/ms191160(v=SQL.105).aspx>

— Add the necessary check constraints. Otherwise you will see the following error.

–Msg 4982, Level 16, State 1, Line 1

–ALTER TABLE SWITCH statement failed. Check constraints of source table ‘PartitionTest.dbo.Order\_Copy’ allow values that are not allowed by range defined by partition 4 on target table ‘PartitionTest.dbo.Order’.

ALTER TABLE Orders\_Copy ADD CHECK (OrdDate> ‘2013-06-30T23:59:59.997’ and OrdDate <= ‘2013-09-30 23:59:59.997’ );

go

checkpoint

go

— Switch the partition that we had earlier swapped out to the Test table back.

ALTER TABLE Orders\_Copy SWITCH TO Orders PARTITION 4;

GO

— Check and will see no logged operations on that table

select Operation,AllocUnitName,count(\*) as NumLogRecords from fn\_dblog(NULL,NULL)

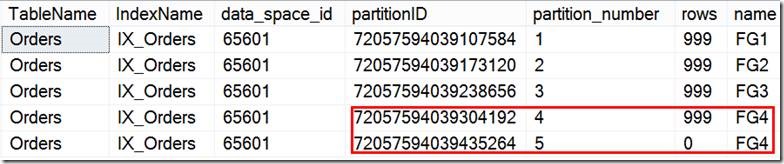
where AllocUnitName= ‘dbo.Orders.IX\_Orders’

group by Operation,AllocUnitName

order by count(\*) desc

[clip_image012](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/24/03/metablogapi/0842.clip_image0123_61ABA320.png)

And viewing the metadata:

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/24/03/metablogapi/3010.clip_image016_3DFC3816.jpg)

Moral of the story:

To avoid data movement during partition split and merge operations, it is always recommended to keep at least the ‘left most’ and ‘right most’ partitions of a partitioned table to be empty. But in cases where it is not possible or if data has been inadvertently populated in to these partitions, you can use the partition switch in/out functionality to avoid data movement during split/merge operations.

As a side note, as far as partitioning goes, the SQL Cat team has a partition management utility that can be used – <http://sqlpartitionmgmt.codeplex.com/>.

[Download Partition Script here](https://skydrive.live.com/?cid=7ceb4fd76f0a9d5d&id=7CEB4FD76F0A9D5D%21116&authkey=!)

Kalyan Yella – Sr. Premier Field Engineer

Denzil Ribeiro – Sr. Premier Field Engineer  ( @DenzilRibeiro )

Tags [Database Engine](https://blogs.msdn.microsoft.com/sql_pfe_blog/tag/database-engine/) [IO](https://blogs.msdn.microsoft.com/sql_pfe_blog/tag/io/) [Performance](https://blogs.msdn.microsoft.com/sql_pfe_blog/tag/performance/) [Transaction Log](https://blogs.msdn.microsoft.com/sql_pfe_blog/tag/transaction-log/)

Comments (9)

[Cancel reply](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/#respond)

Top of Form



Name \* 

Email \* 

Website 







Bottom of Form

https://secure.gravatar.com/avatar/3651e719491c63d4424134d66fe53db9?s=56&d=mm&r=gtobi says:

[August 14, 2013 at 3:13 am](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/#comment-823)

Of course the gold standard would be making SQL Server automatically choose the right split/merge "direction". Always move the (approximate) smallest amount of data, which in this demo was zero.

Any particular reason this is not implemented? It seems like a small effort with high benefit.

[Reply](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/?replytocom=823#respond)

https://secure.gravatar.com/avatar/d72e672ba7d6eb53c07a682ba62651de?s=56&d=mm&r=g[denzilr@outlook.com](http://blogs.msdn.com/denzilr_4000_outlook.com/ProfileUrlRedirect.ashx) says:

[August 14, 2013 at 6:33 am](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/#comment-833)

The decision point currently is only where the boundary is going to fall, for a Range left, the new partition is always created to the left of the split boundary, in range right it is to the right. As far as I know, there is no logic to look at data in that last partition but then you can argue that in a several million row partition that could take time as that last partition can have values from that partition boundary to infinity. I will filing some improvement though, as it would be nice if we did not move data in this case but we currently do. For that matter even when you switch a partition from another table, have to have a constraint indicating the range of data as we don't "check" the data.

For a Merge that is a different situation and there is a fix actually – [support.microsoft.com/…/976379](http://support.microsoft.com/kb/976379)

[Reply](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/?replytocom=833#respond)

https://secure.gravatar.com/avatar/3651e719491c63d4424134d66fe53db9?s=56&d=mm&r=gtobi says:

[August 14, 2013 at 7:47 am](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/#comment-843)

Thanks for the useful response. For the merge case I think SQL Server should use the query processor to execute "select max(partitioning\_col) from T where $partition = x". That might well amount to an index seek returning one row, so it would be extremely fast. Even if it does a CI scan the scan will always be >10x faster than copying the data.

But I guess the responsible PMs have that feature in their lists… I hope it bubbles up soon because the web forums are filled with people having this issue and not knowing why.

[Reply](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/?replytocom=843#respond)

https://secure.gravatar.com/avatar/3651e719491c63d4424134d66fe53db9?s=56&d=mm&r=gtobi says:

[August 14, 2013 at 7:48 am](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/#comment-853)

(And I was talking about the split case, not about the merging).

[Reply](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/?replytocom=853#respond)

https://secure.gravatar.com/avatar/f9e6456071b76565e4f2258fa347ff90?s=56&d=mm&r=gLehighBob says:

[April 9, 2014 at 12:12 pm](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/#comment-1363)

why do the final two partitions, in the last metadata query results have the same partition name of 'FG4'.  They have different partition id's and different partition numbers.  I'd like to use this example, however it's important that my filegroup names describe their contents and not be duplicated.

[Reply](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/?replytocom=1363#respond)

https://secure.gravatar.com/avatar/3651e719491c63d4424134d66fe53db9?s=56&d=mm&r=gFG4 says:

[April 9, 2014 at 1:50 pm](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/#comment-1373)

The name 'FG4' is the name of the File group. And that is showing same file group name for last two partitions because we used 'NEXT USED [FG4] in the Alter Partition Scheme statement. You can create a new file group and use that for your next partition if you want, but the partition and the staging table need to be in the same file group.

[Reply](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/?replytocom=1373#respond)

https://secure.gravatar.com/avatar/3651e719491c63d4424134d66fe53db9?s=56&d=mm&r=gGauravdeep Singh says:

[February 3, 2015 at 4:33 pm](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/#comment-1633)

Excellent article … Thanks for sharing

[Reply](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/?replytocom=1633#respond)

https://secure.gravatar.com/avatar/039c6505d95059a663aef54d7c964009?s=56&d=mm&r=gPeter says:

[February 2, 2017 at 12:09 pm](https://blogs.msdn.microsoft.com/sql_pfe_blog/2013/08/13/oops-i-forgot-to-leave-an-empty-sql-table-partition-how-can-i-split-it-with-minimal-io-impact/#comment-6025)

Hi,  
we have a table with millions of records.  
We are planning to partition it.  
Now we are preparing to create the clustered index on the table (after we already have the partition scheme, etc)

After reading this article, I’m thinking that maybe, we should rather do the switch-split steps (multiple times), you showed in the solution section, n-times, to make the partitions on the table with full of records…. instead of doing it purely by force…. but I’m not sure.