An unexpected error occurred while processing the request

Last updated by | Vitor Tomaz | Oct 1, 2020 at 8:01 AM PDT

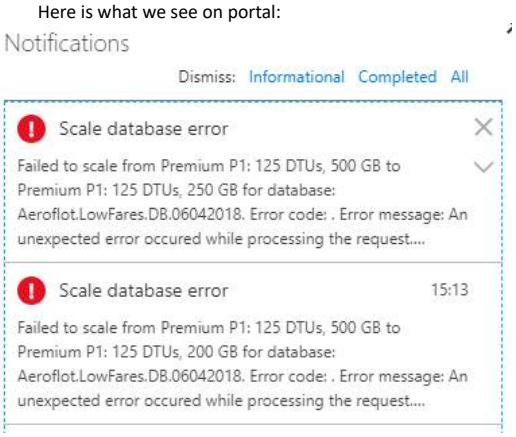
An unexpected error occurred while processing the request

Monday, May 7, 2018
19:42

Our customer got the error message scaling down the database or trying to reduce the size.

We saw several issues that our customers has the data file size very fragmented, unfortunately, the message doesn't report a correct message.

Here is what we see on portal:



Our Azure Product will fix the correct message, working on this defect:
http://sqlbuvsts01:8080/Main/SQL%20Server/_workItems#_a=edit&id=11829072&triage=true

After following this process our customer was able to fix the problem:

PATH 1:

- **STEP 1: Rebuild the indexes.**
  - **ALTERNATIVE 1.1.: If you remove high number of rows of one specific table, you could just only Could you please tell us if you remove data from a specific table? If the answer is YES, execute this query on the specific table:** ALTER INDEX ALL ON dbo.table1 REBUILD;
  - **ALTERNATIVE 1.2.: If you want to review the unused space for each table and run only the ALTER INDEX ALL when you have a high number of unused space in specific tables, please, run the following TSQL:**

```
SELECT
    t.NAME AS TableName,
    s.Name AS SchemaName,
    p.rows AS RowCounts,
    SUM(a.total_pages) * 8 AS TotalSpaceKB,
    CAST(ROUND(((SUM(a.total_pages) * 8) / 1024.00), 2) AS NUMERIC(36, 2)) AS TotalSpaceMB,
    SUM(a.used_pages) * 8 AS UsedSpaceKB,
    CAST(ROUND(((SUM(a.used_pages) * 8) / 1024.00), 2) AS NUMERIC(36, 2)) AS UsedSpaceMB,
    (SUM(a.total_pages) - SUM(a.used_pages)) * 8 AS UnusedSpaceKB,
    CAST(ROUND(((SUM(a.total_pages) - SUM(a.used_pages)) * 8) / 1024.00, 2) AS NUMERIC(36, 2)) AS UnusedSpaceMB
FROM
    sys.tables t
INNER JOIN
    sys.indexes i ON t.OBJECT_ID = i.object_id
INNER JOIN
    sys.partitions p ON i.object_id = p.OBJECT_ID AND i.index_id = p.index_id
INNER JOIN
    sys.allocation_units a ON p.partition_id = a.container_id
LEFT OUTER JOIN
    sys.schemas s ON t.schema_id = s.schema_id
WHERE
    t.is_ms_shipped = 0
    AND i.OBJECT_ID > 255
GROUP BY
    t.Name, s.Name, p.Rows
ORDER BY
    t.Name
```

  - **ALTERNATIVE 1.3: If you want to rebuild all indexes of your database. The link below will give you access to the process that needs to be done:**

https://blogs.msdn.microsoft.com/azuresqldbsupport/2016/07/03/how-to-maintain-azure-sql-indexes-and-statistics/

- Here is the script that you can use to rebuild all indexes/heap tables on the database.

```
-- Rebuild All Indexes
-- This query will rebuild all the indexes on all tables in your database.
-- remove the comments from EXEC sp_executesql in order to have the commands actually rebuild the indexes, instead of just printing the commands.
SET NOCOUNT ON
GO

DECLARE defragcursor CURSOR FOR
SELECT table_schema, table_name
FROM information_schema.tables
    where TABLE_TYPE = 'BASE TABLE'
OPEN defragcursor

DECLARE @tableSchema NVARCHAR(128)(
DECLARE @tableName NVARCHAR(128)
DECLARE @Statement NVARCHAR(300)

FETCH NEXT FROM defragcursor INTO @tableSchema, @tableName

WHILE (@@FETCH_STATUS = 0)
BEGIN
  SET @Statement = 'ALTER INDEX ALL ON '  + '[' + @tableSchema + ']' + '.' + '[' + @tableName + ']' + ' REBUILD'
  PRINT @Statement -- comment this print statement to prevent it from printing whenever you are ready to execute the command below.
  --EXEC sp_executesql @Statement -- remove the comment on the beginning of this line to run the commands
  FETCH NEXT FROM defragcursor INTO @tableSchema, @tableName
END

CLOSE defragcursor
DEALLOCATE defragcursor
GO
SET NOCOUNT OFF
GO
```

- **STEP 2: Shrink the data**

    - The command is DBCC ***SHRINKFILE (1, <filesizemb>);***
    - If the size to shrink is too high, you could use the following script to run in batches of 50 Gb.

Please change within the script the "DesiredFileSize" and "ShrinkChunkSize" at your convenience.

Find this script on GitHub (IncrementalShrink)

```
/*********************************************
Incremental Shrink for data file - Azure SQL
*********************************************/
set nocount on
declare @CurrentFileSize int
declare @DesiredFileSize int
declare @ShrinkChunkSize int
declare @ActualSizeMB int
declare @ErrorIndication int
declare @dbFileID int = 1
declare @lastSize int
declare @SqlCMD nvarchar(max)
declare @MSG nvarchar(100)
/*set this values for the current operation, size is in MB*/
set @DesiredFileSize = 200000
set @ShrinkChunkSize = 50
select @CurrentFileSize = size/128 from sysfiles where fileid=@dbFileID
select @ActualSizeMB = (sum(total_pages) / 128)  from sys.allocation_units
set @msg = 'Current File Size: ' + cast(@CurrentFileSize as varchar(10)) + 'MB'
raiserror(@msg,0,0) with nowait
set @msg = 'Actual used Size: ' + cast(@ActualSizeMB as varchar(10)) + 'MB'
raiserror(@msg,0,0) with nowait
set @msg = 'Desired File Size: ' + cast(@DesiredFileSize as varchar(10)) + 'MB'
raiserror(@msg,0,0) with nowait
set @msg = 'Interation shrink size: ' + cast(@ShrinkChunkSize as varchar(10)) + 'MB'
raiserror(@msg,0,0) with nowait
set @ErrorIndication =
        case
                when @DesiredFileSize > @CurrentFileSize then 1
                when @ActualSizeMB > @DesiredFileSize then 2
else 0 end
if @ErrorIndication=1  raiserror('[Error] Desired size bigger than current size',0,0) with nowait
if @ErrorIndication=2  raiserror('[Error] Actual size is bigger then desired size',0,0) with nowait
if @ErrorIndication=0  raiserror('Desired Size check - OK',0,0) with nowait
set @lastSize = @CurrentFileSize+1
while @CurrentFileSize > @DesiredFileSize /*check if we got the desired size*/ and @lastSize>@CurrentFileSize /* check if there is progress*/ and @ErrorIndication=0
begin
        set @msg = cast(getdate() as varchar(100)) + ' - Iteration starting'
        raiserror(@msg,0,0) with nowait
```

```
        select @lastSize = size/128 from sysfiles where fileid=@dbFileID
        set @sqlCMD = 'dbcc shrinkfile('+cast(@dbFileID as varchar(7))+','+ cast(@CurrentFileSize-@ShrinkChunkSize as varchar(7)) +') with no_infomsgs;'
        exec(@sqlCMD)
        select @CurrentFileSize = size/128 from sysfiles where fileid=@dbFileID
        set @msg = cast(getdate() as varchar(100)) + ' - Iteration completed. current size is: ' + cast(@CurrentFileSize as varchar(10))
        raiserror(@msg,0,0) with nowait
    end
    print 'Done'
```

- This will attempt to shrink the database file down to just under desired space and not actually try and flatten the entire file. This may still take some time to complete but should likely be successful. It is going to maintain any progress, so even if it timeouts out halfway through, that should not be lost. Here is a command you can run to see the total size of the file to confirm it is under the desired space mark:

  SELECT file_id, name as 'db_file', CAST(FILEPROPERTY(name, 'SpaceUsed') AS int)/128.0 as 'size(MB)', size/128.0 - CAST(FILEPROPERTY(name, 'SpaceUsed') AS int)/128.0 as 'allocated_size(MB)', size/128.0 as 'total_size(MB)' from sys.database_files;

- **STEP 3: Run this script to review the DB size**

```
select * from sys.database_files

SELECT (SUM(reserved_page_count) * 8192) / 1024 / 1024 AS DbSizeInMB
FROM    sys.dm_db_partition_stats
```

- **STEP 4: Please go to Azure Portal and try to scale down DB**

How good have you found this content?