

# Increase database size stuck due to conflict operations

Last updated by | Vitor Tomaz | Feb 24, 2023 at 3:24 AM PST

---

## Contents

- [Issue](#)
- [Investigation/Analysis](#)
- [Mitigation](#)
  - [Issue & Manual Mitigation for SQLPackage Import failure \(...\)](#)
- [RCA Template](#)
  - [RCA Only for those hitting SQLPackage Import failure](#)
  - [RCA for operation that recovered itself](#)
- [Internal Reference](#)
- [Root Cause Classification](#)

## Issue

Increase database size operations stuck and eventually timeout.

## Investigation/Analysis

1. Confirm from ASC and kusto for the failed scaling operation details.

```
MonManagementExceptions  
| where request_id == "<issue_request_id>"
```

```
MonManagement  
| where request_id = "<issue_request_id>"
```

2. Check MonSQLSystemHealth, add file error 3023 is returned.

```
MonSQLSystemHealth  
| where AppName == "app_name"  
| where originalEventTimestamp between (datetime(starttime)..datetime(endtime))
```

## Example output

**Message**

2022-05-13 05:48:28.76 spid85 Error 3023: [DynamicDeactivationTask::DoDbOperation]  
DynamicFileAllocationTask failed for database id XXXXXXXXXXXXXXXX

2022-05-13 05:48:28.76 spid85 Call DynamicFileAllocationDetection to grow/add data file for  
database [XXXXXXXXXXXXX] failed,with error (0x00000000)

**Mitigation**

No known mitigation, issue is transient unless seen during SqlPackage Import. If this needs manual intervention (NOT RECOMMENDED), you might need an ICM to involve PG manually change file max size/add files by JITing and adding files on the node.

**Issue & Manual Mitigation for SQLPackage Import failure (PG operation)**

When files are added during SQL Package Import, DFA hits the serialization error due to multiple backup operations happening at the same time. Failures while adding file can lead to SQLPackage import errors like the following: Exception: System.Data.SqlClient.SqlException (0x80131904): The database '<database\_name>' has reached its size quota. Partition or delete data, drop indexes, or consult the documentation for possible resolutions. at System.Data.SqlClient.SqlConnection.OnError(SqlException exception, Boolean breakConnection, Action`1 wrapCloseInAction) at

System.Data.SqlClient.TdsParser.ThrowExceptionAndWarning(TdsParserStateObject stateObj, Boolean callerHasConnectionLock, Boolean asyncClose) at System.Data.SqlClient.TdsParser.TryRun(RunBehavior runBehavior, SqlCommand cmdHandler, SqlDataReader dataStream, BulkCopySimpleResultSet bulkCopyHandler, TdsParserStateObject stateObj, Boolean& dataReady) at

System.Data.SqlClient.TdsParser.Run(RunBehavior runBehavior, SqlCommand cmdHandler, SqlDataReader dataStream, BulkCopySimpleResultSet bulkCopyHandler, TdsParserStateObject stateObj) at System.Data.SqlClient.SqlBulkCopy.RunParser(BulkCopySimpleResultSet bulkCopyHandler)

Problem with SQLPackage import today is lack of resume-ability. Even one failure like the above causes bulk inserts to fail without retry, causing DB to become corrupted. The only option left for the customer is to retry the entire import. Resume-ability of SQLPackage is supposed to be tackled separately as part of the improvement by the DacFx team to support JEDI ADP scenarios. However, the customer would probably need a workaround here, since a retry is likely to hit the same serialization error.

Possible workaround is to allocate all files necessary for the database upfront (when the database is created). This is controlled by a setting called NumOfDataFilesAtCreationTime. You can apply the following server level override to provision as many files necessary at the time of DB creation.

```
Set-ServerPropertyOverride -LogicalServerName "xzc-i-p-clevert-sql-11" -CategoryName 'DatabaseSettings' -Overr
```

**RCA Template****RCA Only for those hitting SQLPackage Import failure**



Import of your database using SQLPackage failed with error message "The database '**<database\_name>**' has reached its size quota. Partition or delete data, drop indexes, or consult the documentation for possible resolutions." It was because during the import, a background task that lazily allocates more file space to databases hit multiple transient failures due to conflicting operations at flight. This background task was added 2 years ago and failures like this generally recover itself and it did in your case as well. However, the reason you received errors was because as of today, SQLPackage is non-resumable, which means any failure during the import causes import to be unusable. In order to help you with your imports, we temporarily pre allocated all the space that might be required by your database upfront. The team that owns SQLPackage is working on making SQLPackage Import more robust to failures (add ETA from owning team) and we are also investigating if the file allocation background task can be prioritized over other operations to not hit size quota errors at all. While we are working to improve this experience for you, we thank you for your patience and apologize for the inconvenience this might have caused.

### RCA for operation that recovered itself

- If the operation succeeded in the end, use below canned RCA Updating max size on your database "**<database\_name>**" was stuck for more than 6 hours, starting at **<start\_time>** and finally succeeding at **<end\_time>**. It was because the max size change invokes a task that usually runs in low priority in the background to change the max size on the database files. Since this task runs in a lower priority as compared to other operations in your database like backups, encryption changes etc., it couldn't make progress due to these conflicting operations at flight at the same time. This is a rare occurrence and usually happens when your database is undergoing another operation that might conflict with alter file size operations. The max size on the files was updated as soon as the other file operations finished and when it was safe for the task to alter the database. This background task was added 2 years ago and failures like this generally recover itself, like it did in your case, without manual intervention. The engineering team is aware of this issue and is investigating if the file allocation background task can be prioritized over other operations. While we are working to improve this experience for you, we thank you for your patience and apologize for the inconvenience this might have caused.
- If the operation time out in the end, use below canned RCA Update operation on your database "**<database\_name>**" timed out after being stuck on a task for 30 mins, starting at **<start\_time>**, timing out at **<end\_time>** and ultimately finishing the operation (time out) at **<timeout\_time>**. This operation timed out because operations have a default time out of 30 minutes. The max size change operation was stuck on a task that usually runs in low priority to change the max size on the database files. Since this task runs in a lower priority as compared to other operations in your database like backups, encryption changes etc., it couldn't make progress due to these conflicting operations at flight at the same time. This is a rare occurrence and usually happens when your database is undergoing another operation that might conflict with alter file size operations. The max size on the files was updated as soon as the other file operations finished and when it was safe for the task to alter the database. This background task was added 2 years ago and failures like this generally recover itself, like it did in your case, without manual intervention. We are unable to pinpoint to the exact operation that conflicted with the update max size operation because we are out of the log retention period. The engineering team is aware of this transient issue while max size change and is investigating if the file allocation background task can be prioritized over other operations. While we are working to improve this experience for you, please note that this is a rare occurrence. We thank you for your patience and apologize for the inconvenience this might have caused.

### Internal Reference

- [Incident 309692416 - database size increase operation stuck at 1%](#) 

- [Incident 184842805 - SQLPackage BACPAC import failing for large DB](#) 
- [Incident 233108329 - DB scaling operation is ongoing\(Stuck\)](#) 

Code fix Work Item: [Bug 732856](#) 

## Root Cause Classification

Cases resolved by this TSG should be coded to the following root cause:

/Azure SQL v3/CRUD/Database/Scale failure

## How good have you found this content?

