SQL Server Database Restore

Documentation

Table of Contents

[1 Introduction 3](#_Toc485850528)

[2 Technical preview 4](#_Toc485850529)

[3 Deployment 5](#_Toc485850530)

[**3.1 Running from another script** 6](#_Toc485850531)

[**3.2 Exceptions in environments** 6](#_Toc485850532)

[**3.3 Optimize TempDB** 7](#_Toc485850533)

[**3.4 Optimize model settings** 7](#_Toc485850534)

[4 After deployment 8](#_Toc485850535)

[**4.1 Direct results** 8](#_Toc485850536)

[**4.2 System databases** 9](#_Toc485850537)

[**4.3 Filesystem** 9](#_Toc485850538)

[**4.4 SQL Agent Jobs** 10](#_Toc485850539)

[**4.5 SQL Agent Jobs schedules** 11](#_Toc485850540)

[**4.6 Database backup retention settings** 12](#_Toc485850541)

[5 Possible problems 12](#_Toc485850542)

[**5.1 Reporting issues** 12](#_Toc485850543)

[6 Change history 13](#_Toc485850544)

# Introduction

This document describes stored procedure covering all required actions to properly restore database to SQL Server instance. Including scenarios where restoring database that is part of Availability Group, and joining it to all secondaries. Restore proccess created based on experience with restoring databases especially in scenarios where restores neeeded on regular basis. Whole restore proccess is described in details further in document. Way of restoring can differ from your own proccess, but that is not definitely bad it is only my approach to doing so and can be wrong, but I have some bullet proof arguments, so feel free to start discussion.

Tested on SQL Server versions >= 2008, so all older versions are not supported and you are running scripts/procedures on your own risk!

# Technical preview

Whole solution consist of two stored procedures, that can be called directly or from SQL Agent job steps. One procedure is needed for all restore scenarios, and another only needed on Availability Group (only AG in further writting) secondary replicas to be able to join database to AG.

* usp\_RestoreDatabase – perform every restore
* usp\_AddDatabaseOnSecondary – only needed on secondary replicas

Both procedures using pure T-SQL approach, I know similar operations can be performed by PowerShell and maybe more efficiently, but I like T-SQL way.

Both procedures cooperating with Ola Halengreen’s maintenance solution procedures (visit here for more details <https://ola.hallengren.com/>), using its ***CommandLog*** table for tracking operations done during restores. Execution of stored procedure will fail if this table not presented in master database and will ask you for deploying it first when you have logging enabled, if not then it is just running 😊 Mentioned *CommandLog* table creation script can be downloaded from [here](https://ola.hallengren.com/scripts/CommandLog.sql).

All important information included in every procedures header also, for example any versions info and small release notes can be found there also

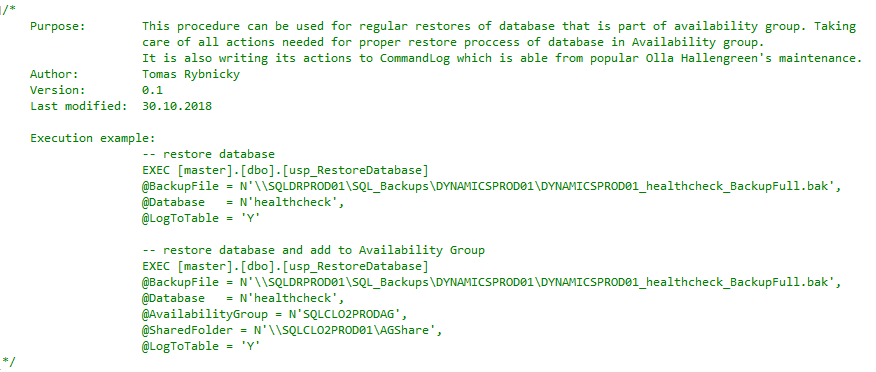


Figure 1 - procedure header info

# Deployment

Only thing you have to do is to copy deplyemnt script from its storage on [GitHub](https://github.com/wetory/SQL-Server-Database-Restore/blob/master/SQL%20Server%20Database%20Restore%20.sql). Copy script to SQL Server Management Studio and run it aganst SQL Server instance you are connected to or use multiquery from Registered Servers. Running script using multi-query is especially benefical when creating procedures on AG replicas, you will avoid unnecesarry clicking when connecting to every replica and running one by one.

You can also change script body if there is something that you do not like there, but only by your own responsibility!

Deplyment script is doing nothing magical, only creating two stored procedures in master database.Firts it checks if stored procedure already exists and drop and re-create it. **So please be aware of overwriting of your procedure if you have same namming.** You can also find some important info in script header, reather to read it before running anything against your servers, you should be aware of what you are doing also on non-production servers.

# After deployment

## **Direct messages & results**

After proper execution you can check messages for detailed steps which have been done over instance and also for possible related error messages.

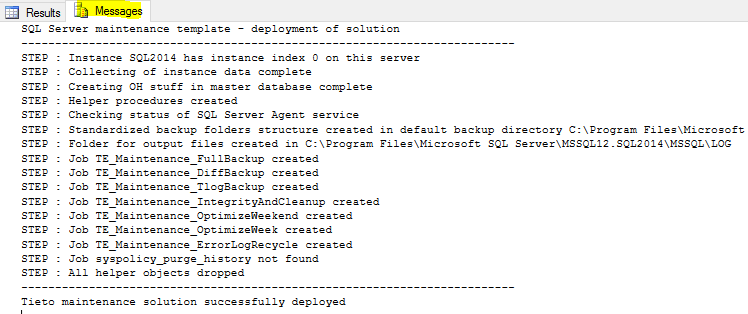


Figure 5 Script messages

And in some cases you may be asked for schedules or you can use it for self-review of schedules for deployed jobs or you can attach to install change etc. there is result table with job names and schedules.

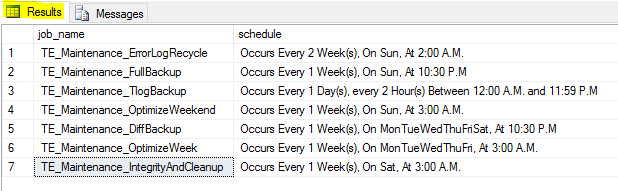


Figure 6 Script results

## **Stored procedures**

You can also can see tempdb and model database file properties that it was changed if parameter **@OptimizeTempdb** is set to Y (what means YES).

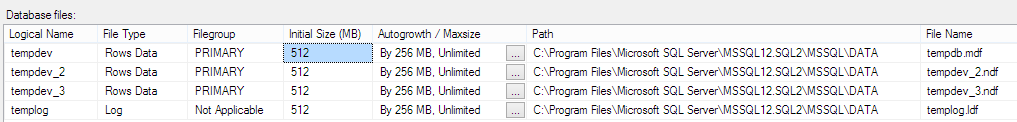


Figure 7 TempDB files optimization (3 CPUs)

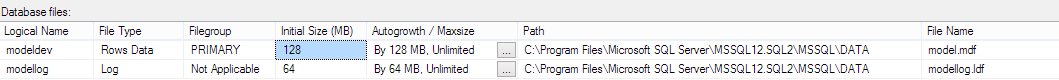


Figure 8 Model DB optimization

## **Filesystem**

Then on file system of the hosting server backup folders structure is created in instance default backup location (can be found in instance properties or in the registry of the hosting server). Backup folders structure is inherited from DMS tool as this is replacement so we want to have same structure also.

## **SQL Agent Jobs**

This section describes SQL Agent jobs deployed to the target SQL Server instance by executing mentioned script. All jobs are part of SQL Server maintenance since deployed and all colising Maintenance plans or user defined SQL AGen jobs should be disabled or removed if there are any to prevent duplicate maintenance or backup tasks done.

Following list contain short description for all steps in all jobs that are part of solution.For more info about procedures executed in steps, read [offical Ola Halengreen’s documentation](https://ola.hallengren.com/).

Every step also contains link to official Microsoft documentation of SQL Server maintenance tasks and related command and procedures used to run them.

**TE\_Maintenance\_DiffBackup**

* **TRACEON 3042** – enabling incremental backup file growth <https://msdn.microsoft.com/en-us/library/ms188396.aspx>
* **Full Backup - System databases** – full backup of all system databases <https://msdn.microsoft.com/en-us/library/ms186289.aspx>
* **Diff Backup - User databases –** differential backup of all user databases <https://msdn.microsoft.com/en-us/library/ms175526.aspx>
* **TRACEOFF 3042 –** disabling incremental backup file growth <https://msdn.microsoft.com/en-us/library/ms188396.aspx>
* **Integrity Physical** **- All databases –** DBCC CHECKDB with option PHYSICAL\_ONLY <https://msdn.microsoft.com/en-us/library/ms176064.aspx>

**TE\_Maintenance\_ErrorLogRecycle**

* **Cycle Error Log –** close existing SQL Log and create new one <https://technet.microsoft.com/en-us/library/ms182512(v=sql.110).aspx>

**TE\_Maintenance\_FullBackup**

* **TRACEON 3042** – enabling incremental backup file growth <https://msdn.microsoft.com/en-us/library/ms188396.aspx>
* **Full Backup - System databases**– full backup of all system databases <https://msdn.microsoft.com/en-us/library/ms186289.aspx>
* **Full Backup - User databases**– full backup of all user databases <https://msdn.microsoft.com/en-us/library/ms186289.aspx>
* **TRACEOFF 3042–** disabling incremental backup file growth <https://msdn.microsoft.com/en-us/library/ms188396.aspx>

**TE\_Maintenance\_OptimizeWeek**

* **Index Optimize –** rebuilding or reorganizing indexes based on their fragmentation (0% -30%, 30% - 60%, 60% - 100%). Excluding offline index rebuilds to prevent table locks.

<https://technet.microsoft.com/en-us/library/ms190910(v=sql.105).aspx>

* **Update Statistics –** updating of query optimization statistics on a table or indexed view <https://msdn.microsoft.com/en-us/library/ms187348.aspx>

**TE\_Maintenance\_OptimizeWeekend**

* **Index Optimize –** rebuilding or reorganizing indexes based on their fragmentation (0% -5%, 5% - 30%, 30% - 100%). Including offline index rebuilds.

<https://technet.microsoft.com/en-us/library/ms190910(v=sql.105).aspx>

* **Update Statistics –** updating of query optimization statistics on a table or indexed view <https://msdn.microsoft.com/en-us/library/ms187348.aspx>

**TE\_Maintenance\_IntegrityAndCleanup**

* **Integrity Check –** DBCC CHECKDB with full scans over all objects in databases <https://msdn.microsoft.com/en-us/library/ms176064.aspx>
* **Command Log Cleanup –** removing records older than 30 days from [master].[dbo].[CommandLog] table
* **Output Files Cleanup -** removing files older than 30 days from output files folder (<<log directory>> + \TE\_Maintenace\_OutputFiles\)
* **History Cleanup -** removing records older than 30 days from system tables

**TE\_Maintenance\_SyspolicyPurgeHistory (just renamed syspolicy\_purge\_history job)**

* **Verify that automation is enabled.**
* **Purge history.**
* **Erase Phantom System Health Records.**

**TE\_Maintenance\_TlogBackup**

* **TRACEON 3042**– enabling incremental backup file growth <https://msdn.microsoft.com/en-us/library/ms188396.aspx>
* **Tlog Backup - System databases –** backup of transaction log for all system databases (in full recovery model)

<https://msdn.microsoft.com/en-us/library/ms191429.aspx>

* **Tlog Backup - User databases –** backup of transaction log for all user databases (in full recovery model)

<https://msdn.microsoft.com/en-us/library/ms191429.aspx>

* **TRACEOFF 3042–** disabling incremental backup file growth <https://msdn.microsoft.com/en-us/library/ms188396.aspx>

All suggested SQL Server database maintenance tasks are pretty well described at this site <https://technet.microsoft.com/en-us/library/ms140255(v=sql.105).aspx>. This maintenance solution is trying to find some accurate way to meet all database needs and include standardization definitions and schedules for use in company which is taking care of many SQL Server environments and want to keed all instances in good health condition.

## **SQL Agent Jobs schedules**

All schedules are set up during deployment as agreed standardized and optimal schedules for related SQL Server maintenance task. If SQL Server instance require different schedules because of some reason it can be modified after deployement same as whole maintenance solution **can be (and also should be) modified** to fit exact needs of SQL Server instance it is deployed to.



Figure 10 - SQL Agent jobs schedules

## **Database backup retention settings**

Following drawing is describing available retore times that are feasible when standardized SQL Server maintenance is used for taking care of database backup reoutines.

Drawing does not describe point-in-time restores for databases in full recevory model. Backup of transaction log is running every 1 hour by default (can be adjusted for frequently changed databases)

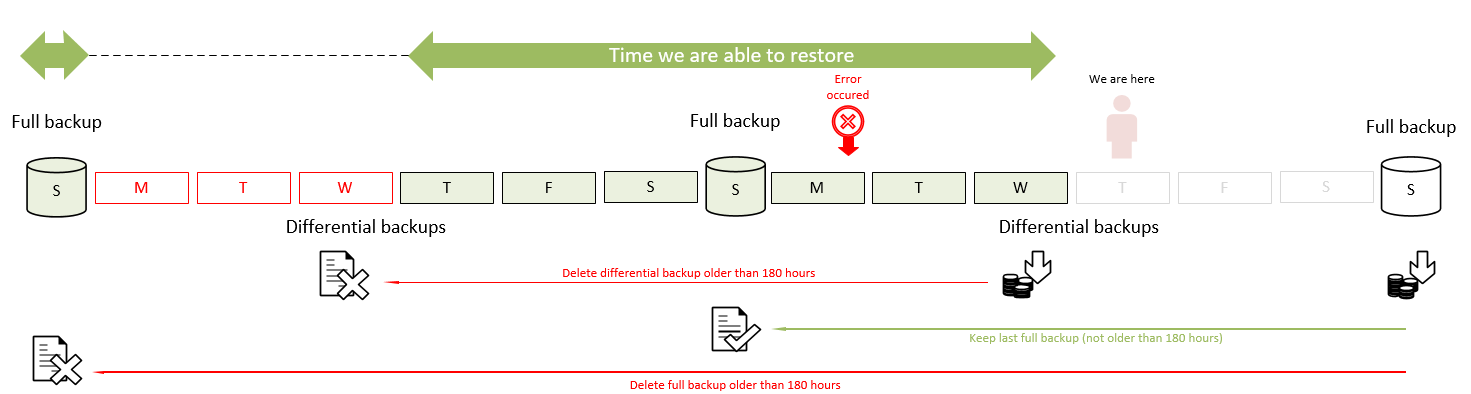


Figure 11 - backup retention settings

# Possible problems

There was testing of the solution ongoing for several weeks for debugging and tuning purposes and all known problems has been fixed already, but as everything also this script can cause some issues in different environments.

I’m assuming only following possible issues:

* problems with old backup files removal
* problems with compression on not supported SQL versions/editions (partially fixed)

And some other possible problems can be related to OH stuff in the solution so, please be so kind and try to check this FAQ <https://ola.hallengren.com/frequently-asked-questions.html> first before asking me directly.

## **Reporting issues**

Please report all found issues, current version of the solution is the first one and require some debugging to be “perfect”. Here are some contacts you can write to via email or LYNC:

* [tomas.rybnicky@tieto.com](mailto:tomas.rybnicky@tieto.com) (T-SQL)

# Change history

| Version | Date | Author | Approved by | Change history |
| --- | --- | --- | --- | --- |
| V1.1 | 06.04.2016 | Tomáš Rybnický | Daniel Rojíček | First version of this document |
| V1.2 | 11.05.2016 | Tomáš Rybnický | Daniel Rojíček | Documentation updated |
| V1.3 | 19.05.2016 | Tomáš Rybnický | Daniel Rojíček | Automation support |
| V1.4 | 03.01.2017 | Tomáš Rybnický | Tomáš Rybnický | Scope of document changed |
| V1.5 | 11.01.2017 | Tomáš Rybnický | Tomáš Rybnický | Major updates optimization of SQL Server |
| V1.6 | 14.03.2017 | Tomáš Rybnický | Tomáš Rybnický | Backup retention settings |
| V1.7 | 15.03.2017 | Tomáš Rybnický | Tomáš Rybnický | Job description as editable parameter |