Standardized MS SQL Maintenance

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)

Official documentation of one-script maintenance solution

# Introduction

This document describes maintenance solution used for keeping SQL Server in good condition purposes as agreed maintenance standard used on all SQL Server environments which is deployed after SQL Server instance installation. By default it is configured on agreed schedules but can be manually customized after deployment to the instance directly in SQL Agent jobs.

All environment specific need can be meet by modifications to standard implementation, for more info regarding to customization of Ola Halengreen’s scripts can be found at [official sites](https://ola.hallengren.com/).

Maintenance can be deployed to SQL Server versions >= 2005 (SQL Server 2000 not supported but some solution is in long term plans to have one script maintenance solution as this one)

# Technical preview

Whole solution is mostly based on popular Ola Hallengren’s maintenance procedures (visit here for more details <https://ola.hallengren.com/>). Olla’s procedures have been modified so as to fit needs described in standard specified in our team. Script is creating maintenance jobs as its main objection. All tasks performed on the target SQL Server instance are listed below.

Script perform following stuff:

* create OH procedures in master database
* create and drop helper procedures in master database
* create SQL Agent jobs (full backup, differential backup, log backup, integrity check and index optimize, recycle error log)
* schedule SQL Agent jobs based on agreed standard times
* rename maintenance job syspolicy\_purge\_history
* optimize tempdb according to best practices (this is optional)
* change initial size and Autogrowth on model and tempdb database based on our agreed standard

Some of above described actions are not executed automatically but can be fired by changing input parameters, for more info see section Deployment

All important info also included in script header, containing also some short update history.

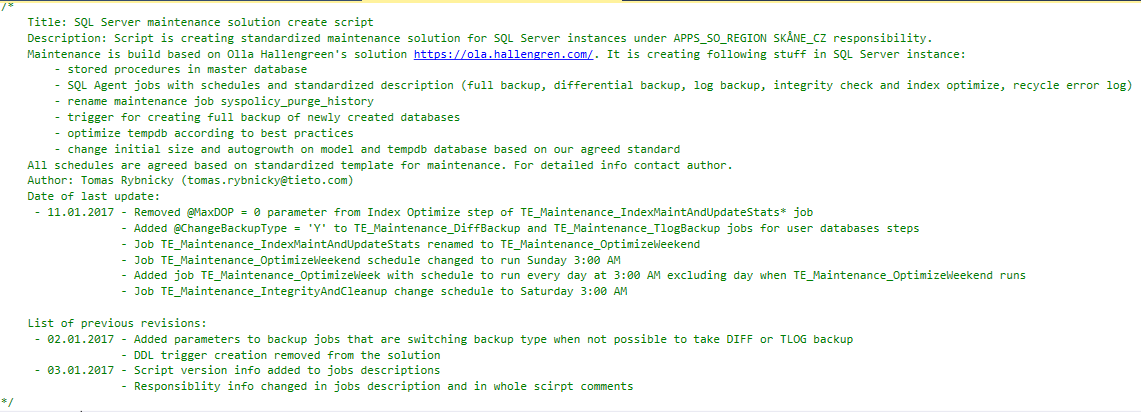


Figure 1 - script header

# Deployment

Only thing you have to do is to copy script from its official storage on [Google](http://spintra.int.tieto.com/teams/eis/service_delivery/bidbas/Shared%20Documents/Automation/Automation%20scripts/TE_Maintenance.sql) [Drive](https://drive.google.com/open?id=0B-2OaDD87adKdm93M3JNSmRmTjg). Copy script to SQL Server Management Studio and set parameters if needed, by default parameters are set as following (all of them are pretty self-commented). You can change them based on environment you are deploying to.

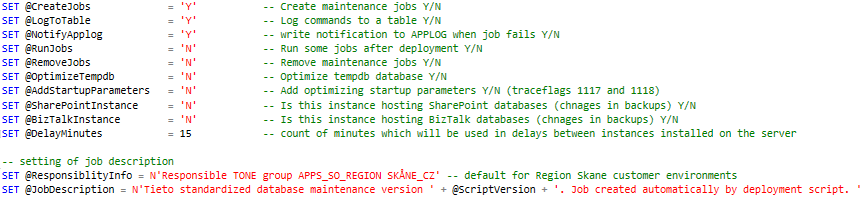


Figure 2 Script parameters

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

You can deploy to single server or to more servers using registered servers, where script will be executed in parallel on all registered instances (never mind if running on one or more servers).

One important thing is that script is creating delayed schedules for more instance running on same machine. Because of reducing system resources load during run at same time on all instances. You can set delay by parameter **@DelayMinutes**. Then every installed instance is given index based on value in registry key **HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Microsoft SQL Server** under value of key name **IntalledInstances.**

## **Running from another script**

From V1.3 there is support for parametrizing run of the script for executing it from another script (batch, PowerShell…). There need to be possibility to set mentioned parameters for execution from “outside”. Any of the parameters are required for running the script, if no value for parameter given default values from declaration will be used. See below screen for more info about possible parameters

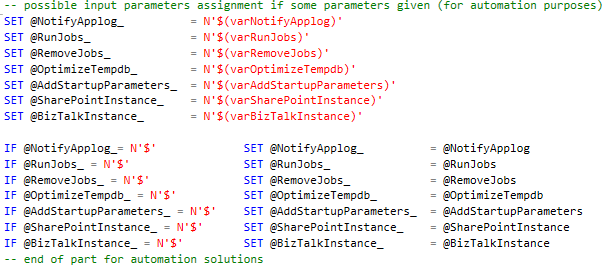


Figure 3 Parameters for "outside" execution

## **Exceptions in environments**

**Parameters:** @SharePointInstance, @BizTalkInstance

Some application specific SQL environments requires some exception in SQL maintenance, for example BizTalk related SQL databases are handled by backups from BizTalk application. Or SharePoint databases can be handled by 3rd party tool like DocAve. There are prepared switch parameters at the script beginning to modify maintenance accordingly.

Now there are only two exceptions for mentioned BizTalk and SharePoint environments as described below.



Figure 4 Environment exception flags

If instance is marked as **@SharePointInstance** then only system databases are handled by this maintenance solution. User databases are handled by DocAve in this situation.

If instance is marked as **@BizTalkInstance** then exceptions for standard BizTalk databases are included in backup step of user databases. Then parameter in script is set as following

*@Databases = 'USER\_DATABASES, -BAMPrimaryImport%, -BAMAlertsApplication%, -BAMAlertsNSMain%, -BizTalkDTADb%, -BizTalkMgmtDb%, -BizTalkMsgBoxDb%, -BizTalkRuleEngineDb%, -SSODB%, -BAMStarSchema%, -BAMAnalysis%, -BAMArchive%, -BizTalkAnalysisDb%'*

This exclusion is done based on official BizTalk documentation <https://msdn.microsoft.com/en-us/library/aa578342.aspx>

## **Optimize TempDB**

**Parameters:** @OptimizeTempdb

There are some optimization recommendations regarding TempDB performance. Based on MS best practices following TempDB configurations should be followed for optimal performance of TempDB database:

* database files should be stored on separate disk drives to provide exclusive IO operations just for communication with this database on disk drives (this is not object of maintenance solution and should be done by design of environment)
* database files should be divided into number of files equally to count of logical processors but maximally up to 8, then advantages of parallel accessing become disadvantage because of internal management of SQL engine
* database auto growth and initial size should be the same on all files mentioned in previous step. We have chosen following values for our standard. Initial size is set to **512 MB**  and file growth is set to **256 MB** which are agreed values not causing any overload when performing auto growth of files and similar issues

.

More info about TempDB optimization can be found on official MS web pages <https://msdn.microsoft.com/en-us/library/ms190768.aspx>.

## **Optimize model settings**

The **model** database is used as the template for all databases created on an instance of SQL Server. Because **tempdb** is created every time SQL Server is started, the **model** database must always exist on a SQL Server system. The entire contents of the **model** database, including database options, are copied to the new database. Some of the settings of **model** are also used for creating a new **tempdb** during start up, so the **model** database must always exist on a SQL Server system.

We are changing model database configuration to more efficient and optimal values in our standardized solution. We are changing initial file size and file growth attributes of this database from default values to agreed values providing some performance improvements on newly created databases with default configuration in comparison to default values set after installation of SQL Server instance. Values change described in table below.

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Default value** | **Changed value** |
| Autogrowth for data file | By 1 MB, Unlimited | By 128 MB, Unlimited |
| Autogrowth for log file | By 10 percent, Unlimited | By 64 MB, Unlimited |
| Initial size for data file | 5 MB | 128 MB |
| Initial size for log file | 2 MB | 64 MB |

Table model configuration

For more info about model database see official MSDN article <https://msdn.microsoft.com/en-us/library/ms186388.aspx>.

# After deployment

## **Direct results**

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

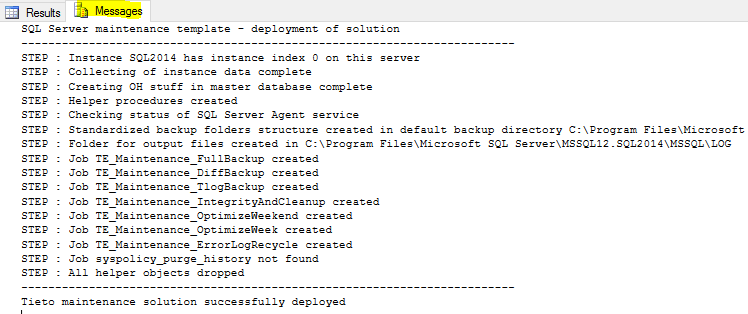


Figure 5 Script messages

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.

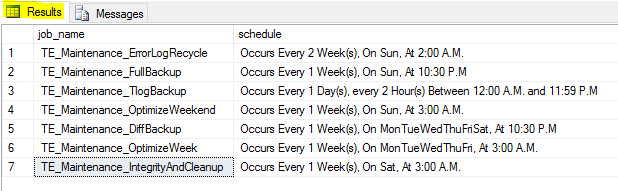


Figure 6 Script results

## **System databases**

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).

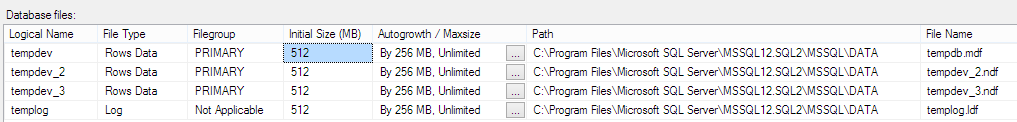


Figure 7 TempDB files optimization (3 CPUs)

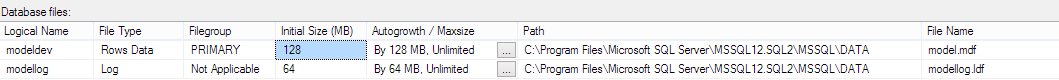


Figure 8 Model DB optimization

## **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.

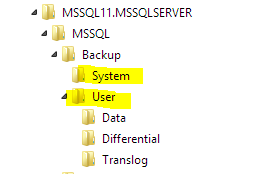


Figure 9 Backup folders structure

## **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)

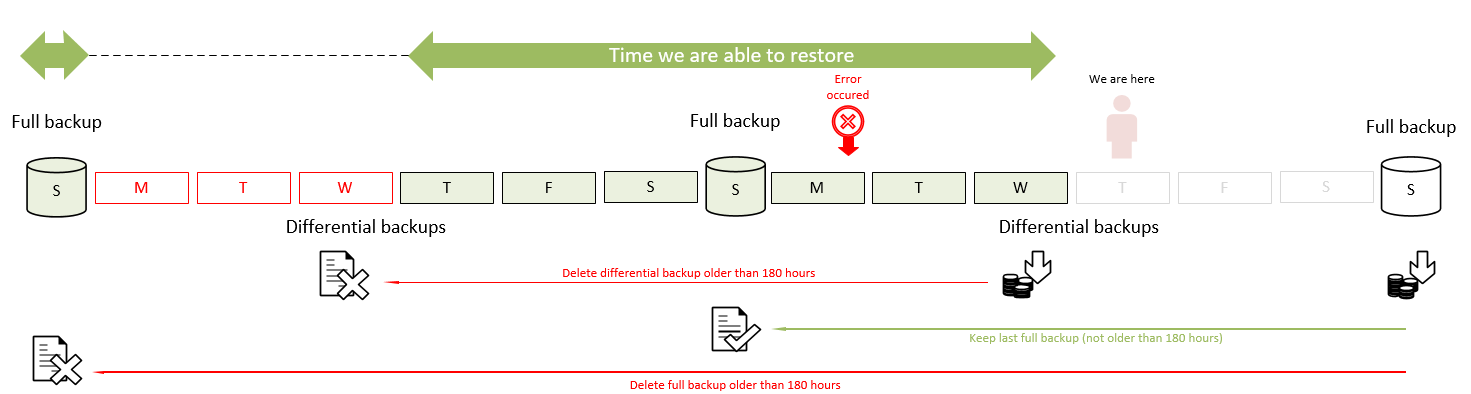


Figure 11 - backup retention settings

# 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 |