



*The World's Largest Community
of SQL Server Professionals*

DBA 101: Best Practices All DBAs Should Follow

Brad M. McGehee
Microsoft SQL Server MVP
Director of DBA Education
[Red Gate Software](#)
www.bradmcgehee.com

My Assumptions About You

- You may be a part-time or full-time DBA
- You may be a DBA Administrator or DBA Developer
- You have less than one year's experience as a SQL Server DBA

- If you have been a DBA for one or more years, then you probably are already familiar with most of this content. On the other hand, maybe you may pick up a tip or two, or may be reminded about something you need to do, but have forgotten.

What We are Going to Learn Today

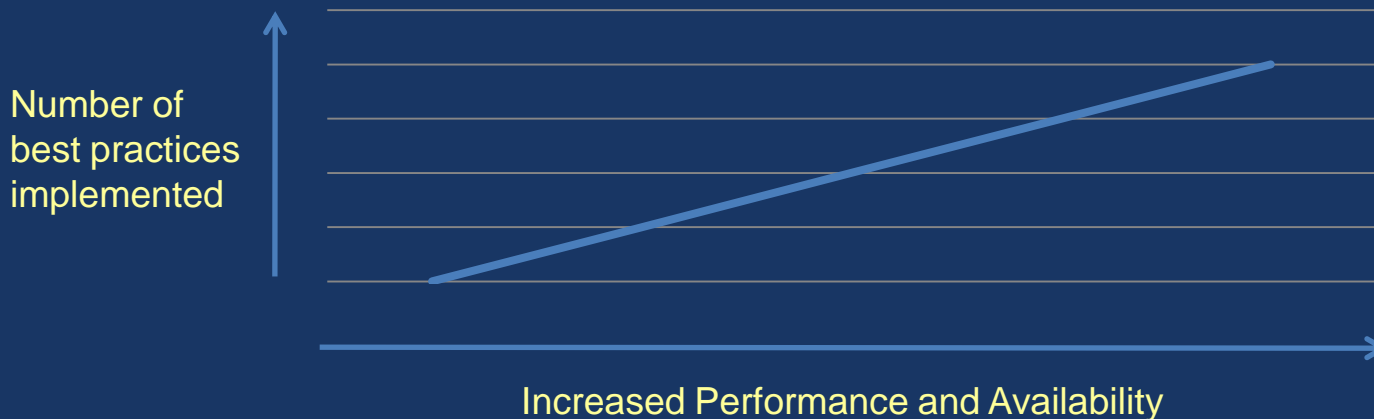
- Benefits of best practices
- Common best practices all DBAs should follow
- Our focus today is on what to do, not how to do it.
- These best practices are just the basics, there are a lot more to do and learn.
- These best practices are based on many of the common mistakes I see DBAs make.
- There are always exceptions to every rule, and not every recommendation discussed here may fit your environment.

Benefits of Focusing on Best Practices

- Helps to optimize SQL Server performance
- Helps to maximum SQL Server availability
- Helps you to be proactive, not reactive
- Helps you to reduce being in “crisis” mode

Everything Counts

- While many of the best practices I discuss today might seem small in scope, the *accumulative effect* of following each and every recommendation can be huge.
- By following these best practices, SQL Server *performance* and *availability* can be boosted by 10%, 20%, or even more.



Check & Verify

- Installing & Upgrading SQL Server
- General Server Configuration
- Memory Configuration
- User Data and Log File Management
- Tempdb Management
- Database Configuration Settings
- Configuring Jobs—General
- Create Index Rebuilding/Reorganize Job
- Create Data Corruption Detection Job
- Set Up Alerts for Critical Errors
- Security Basics
- Implement a Backup/Restore Strategy
- Create a Disaster Recovery Plan
- Test Everything

Installing & Upgrading SQL Server

- Generally, when **installing** a new SQL Server instance:
 - Use the newest OS version with latest SP.
 - Use the newest SQL Server version with latest SP.
 - Test, and once stable, be wary of making changes.
- Generally, when **upgrading** an existing SQL Server instance:
 - Don't upgrade unless you have a really good reason to upgrade. If your instance is working well, don't mess with it.
 - Only upgrade if you need new features or have problems with an old installation, or need to upgrade hardware.
 - It is always safer to upgrade to a new server with a fresh installation of the OS and SQL Server than to upgrade in place.
 - This allows you to test more effectively, and also gives you a backout option.

General Server Configuration

- Ideally, SQL Server instance should be a stand-alone server with no other apps running on it.
- Avoid multiple instances unless you have a really good reason to use them. Consider virtualization instead.
- Unnecessary SQL Server services should be uninstalled or turned off.
- Ideally, don't run antivirus/antispysware software locally.
 - If feasible, try to run in remotely.
 - Or if you have to run locally, exclude MDF, LDF, BAK, and TRN files.

Memory Configuration

- Ideally, use 64-bit hardware and the 64-bit version of the OS and SQL Server.
- If using 32-bit version, and if using 4 GB or more of RAM, ensure that /3GB switch and AWE memory are correctly configured. Correct settings depend on available RAM.

Data and Log File Management

- Remove physical file fragmentation before creating new MDF or LDF files.
- When creating new MDFs and LDFs, pre-size them to minimize autogrowth.
- MDF files should be located on their own volume.
- LDF files should be located on their own volume.
- BAK and TRN backup files should be located on their own volume.

Instant File Initialization

- Enable instant file initialization, prevents MDF files from being zeroed out, saving time when they are created.
- Speeds up CREATE DATABASE, ALTER DATABASE, RESTORE DATABASE, Autogrowth.
- Requires SQL Server 2005/2008, and Windows Server 2003/2008.
- Instant file initialization is turned on if the SQL Server (MSSQLSERVER) service account has been granted the SE_MANAGE_VOLUME_NAME permission. Members of the local Windows Administrator group automatically have this right.

Tempdb Management

- Pre-size files so autogrowth doesn't have to happen often (8MB is default).
- Set autogrowth to avoid many growth spurts, use a fixed amount that minimizes autogrowth use. (10% is default).
- If very active, locate it on its own volume.
- If very active, consider dividing the tempdb into multiple physical files so that the number of files is about 50% to 100% of the number of CPU cores your server has. Each physical file must be the same size.

Database Configuration Settings

- Auto Create Statistics: On
- Auto Update Statistics: On
- Auto Shrink: Off
- Page Verify: Use Checksum (2005/2008), don't turn off
- Autogrowth: Leave on. Use mainly for catching mistakes. File growth should be managed manually. Use fixed amount that minimizes autogrowth occurrences.
- Production databases should be set to FULL RECOVERY so transaction log backups can be made.

Configuring Jobs—General

- Schedule jobs so they don't interfere with production.
- Try to prevent jobs from overlapping.
- Set alerts on jobs so you are notified if they fail.
- Don't schedule automatic shrinking of databases. If you must shrink a file, do it manually and off hours, then rebuild indexes.

Create Index Rebuilding/Reorganize Job

- Indexes need to be rebuilt or reorganized regularly to minimize fragmentation and reduce wasted space.
- Consider *rebuilding* an index if it is heavily fragmented (>30%). In Enterprise Edition, can perform online. In Standard Edition, consider it an off-line job. This **automatically updates statistics**, so you don't need to do this again.
- Consider *reorganizing* an index if it is not heavily fragmented (>5% and <= 30%). This is an online operation and doesn't use a lot of resources. **You must update statistics** afterwards, as this is not automatically done for you.
- Ideally, you should *only rebuild or reorganize indexes that need it*. Use `sys.dm_db_index_physical_stats` to identify what tables/indexes need to be rebuilt/reorganized.

Sample Maintenance Scripts

- Sample script to identify indexes that need to be defragged, and how they are to be defragged.
- <http://ola.hallengren.com/>
- <http://sqlfool.com/2009/06/index-defrag-script-v30/>
- http://www.grics.qc.ca/YourSqlDba/index_en.shtml

Create Data Corruption Detection Job

- Ideally, run DBCC CHECKDB as often as you run full backups.
- Create an appropriate job to run this (or similar) command:

```
DBCC CHECKDB ('DATABASE_NAME') WITH NO_INFOMSGS,  
ALL_ERRORMSG;
```

Note: Consider using PHYSICAL_ONLY option for large or busy production servers to reduce run time.

- Review results to look for problems. If you have a problem, you want to find it as soon as possible to reduce the risk of data loss. **Don't use the repair option.**

Set Up Alerts for Critical Errors

- Create a SQL Server Event Alert for all events with a severity of 19 [fatal] and higher.
- Have alerts sent to you or whoever is responsible for day-to-day monitoring.
- Consider a third-party alerting tool if SQL Server Alerts doesn't meet all of your needs.

Security Basics

- Don't give users more permissions than they need to perform their job. (Critical. Sounds simple, often hard.)
- Don't use the SA account for anything. Assign it a complex password, and keep it handy just in case. Use an account that is a member of the sysadmin role.
- Don't allow an application to use SA or a sysadmin account to access SQL Server.
- Use Windows Authentication security whenever possible.
- Don't give vendors sysadmin access to your servers.
- Log off your SQL Server when done.

Implement a Backup/Restore Strategy

- Create a job to perform full backups daily on **all system and user production databases, plus log backups** hourly (or similar variation).
- Backup using RESTORE WITH VERIFYONLY to help verify backup integrity.
- Set up an appropriate backup retention policy.
- Store backups securely and off-site (not on same disk array or SAN).
- If you have a limited backup window, or have limited disk space, use backup compression. Can be a big time saver.

Create a Disaster Recovery Plan

- You must create a document that outlines, **step-by-step**, in great detail, how you will recover your SQL Servers in the case of any problem, small or large.
- You need to **practice** using the plan so you are familiar with it and can easily implement it.
- Keep Microsoft SQL Server's Product Support phone number handy. Paste it near your computer.
- Remember: **Most "disasters" are small**, such as a corrupted database. Big "disasters" occur very rarely, if ever. But you need to be prepared for both.

Test Everything

- Before you make *any change* on a production SQL Server, be sure you test it first in a test environment.
 - NO EXCEPTIONS!
 - I mean it!
 - Really!
 - No kidding.
 - I wouldn't lie to you.
 - You don't want to lose your job.
 - You'd be crazy not listening to this advice.
 - Civilization as we know it may lie in your hands.

On-Going Tasks

- Monitor MDF and LDF file growth
- Monitor Free Space
- Monitor SQL Server and OS Logs
- Monitor Jobs
- Test Backups
- Monitor Performance
- Analyze Indexes
- Document All Changes

Monitor MDF and LDF File Growth

- MDF and LDF files should be managed manually.
- To do this well, you must monitor file growth and be aware of how much data is being added.
- Be proactive. Plan early to add more space if needed, don't wait and let autogrow do it for you.
- How to perform:
 - In SQL Server 2005, use “Standard Reports” to help you monitor this.
 - In SQL Server 2008, use Performance Data Collector to help you monitor this.
 - Use third-party monitoring tool.

Monitor Free Space

- Try to keep your disks below 85% of their capacity.
- How to perform:
 - Check this manually and record results
 - Use Performance Monitor and set up an alert
 - Use a third-party tool

Monitor SQL Server and OS Logs

- Daily, review OS Event Logs
 - Application
 - System
 - Security
- Daily, review SQL Server Logs
- How to perform:
 - Manual check using Event Viewer or SSMS
 - Have alerts e-mailed to you from SQL Server Agent or Event Viewer
 - Use third-party tool

Monitor Jobs

- Daily, (or more often) verify that all jobs have run successfully.
- How to perform:
 - Set up job alerts
 - Manually check
 - Have SQL Server Agent send you a message upon failure
 - Use third-party tool

Monitor Alerts

- If you do set up any of the many available alerts, be sure you look at them.
- Sometimes, it is easy to be complacent and to ignore alerts.
- The secret is to keep alerts to a minimum by carefully creating the alerts, setting them up so that only the most critical alerts are sent.

Test Backups

- At the very minimum, perform a test restore on at least one key production database, per SQL Server instance, weekly.
- If you really want to seek perfection, restore every backup to a test server, then run DBCC CHECKDB on the restore. If this works, you can be assured you have a good backup.

Monitor Performance

- Monitor performance regularly, daily if necessary.
- Identify potential problems early before they become critical problems.
- How to monitor:
 - Performance Monitor
 - Profiler
 - DMVs
 - SQL Server 2005 Performance Dashboard
 - SQL Server 2008 Performance Data Collector
 - Third-Party Tools

Analyze Indexes

- Indexing needs of a database can change over time as data changes, and as how data is queried changes.
- Every 1-3 months, perform a Profiler trace on your production servers, then use the trace as the source for the Database Engine Tuning Advisor to help you identify missing indexes, indexed views, along with identifying unused indexes that can be deleted.

Take Homes for Today

- By focusing on the basics, you gain the following:
 - Better SQL Server **performance**
 - Higher SQL Server **availability**
 - Being **proactive** helps to you prevent being in a “crisis” mode all the time
 - You become a better, more **professional** DBA
- The total effect of following each and every recommendation made today **can be huge**.
- What you learned today is only the **tip of the iceberg**, you will need to take time to learn many other best practices.
- When you get **back to work**, use this as a checklist to give your SQL Servers a quick health check.

Q&A

- Please ask your questions clearly and loudly.
- If you don't get your questions answered now, see me after the session, or e-mail me.

Find Out More

Free E-Books:

- www.sqlservercentral.com/Books

Check these out:

- www.SQLServerCentral.com
- www.Simple-Talk.com

Contact me at:

bradmcgehee@hotmail.com

Blogs:

www.bradmcgehee.com

www.twitter.com/bradmcgehee

[Click Here for a free 14-day trial of the Red Gate SQL Server Toolbelt](#)