StraightPath Solutions

SQL SERVER CONFIGURATION BEST PRACTICES

Considerations Before, During, and After Installation

Straight Path IT Solutions

www.straightpathsql.com

SQL SERVER CONFIGURATION BEST PRACTICES

TABLE OF CONTENTS

INTRODUCTION

There are many tips on the Web about installing SQL Server. Some are more helpful than others. The one thing that seems to be universally true, though, is "Next-Next-Next-Finish" installations with no best practices setup end up with issues at some point. Microsoft has slowly worked to improve this over the years by improving defaults and help for those installing SQL Server, but there is still a long way to go.

This guide encompasses some of the collective wisdom from the Straight Path team as advice for things to consider before, during and after installing SQL Server. This list goes a long way to ensuring a secure, reliable and highly performing SQL Server instance, and it brings you a long way towards one of our SQL Server health checks showing a healthy SQL Server. (https://straightpathsql.com/sql-server-health-check/)

Read on and start your next SQL Server installation off right!

SQL SERVER CONFIGURATION BEST PRACTICES

BEFORE YOU START

Best practices for SQL Server installation start long before you even buy your licenses or configure your virtual or physical machines. Here are some things we like to consider before even beginning an installation of SQL Server with a client.

Storage

- Drives should be the correct RAID level. You want to optimize for performance AND reliability. With modern SANs, you may have less choice, but don't install SQL Server on striped only or put your log files on RAID 5 when RAID 10 is available.
- Size the OS drive at least 100gb. This gives room for dumps, for growth events and for scale. Running out of space on the OS drive is embarrassing. It isn't just a "SQL problem; it affects everything.
- Design multiple paths to your disks. Applications like to rip through data and having multiple paths from the server to the switches to the storage devices helps.
- Size storage to support at least two years of estimated usage and growth. Go big or go home (or go back to your storage team repeatedly).

Network

- Consider teaming physical NICs to give redundancy and improved throughput.
- Design multiple paths for networks also. Build for redundancies.

Availability & Recovery

- Design backups to support your recovery point objective and recovery time objective requirements. This means you need to sort out your requirements ahead of time. Don't just guess and hope it works. See more about this in our blog post, You Can Restore It! (right?) (https://straightpathsql.com/archives/2011/02/can-you-restore-sql-server-sqlu/)
- Consider high availability and disaster recovery needs up front, before you decide on licensing.

Licensing & Hardware

- Strongly push for a dedicated server. Production SQL Servers should host SQL Server only – not apps, and not even related services like SSRS, SSIS, or SSAS.
- Evaluate license needs and consider needed features. Since SQL Server 2016 SP1, standard edition has included a lot of enterprise-level features, including partitioning, columnstore, in-memory OLTP, limited clustering and availability group support, backup compression, and support for up to 16 processor cores and 128GB of RAM. While this may satisfy the needs of some enterprises, others will require Enterprise Edition in order to leverage features like online index operations, multi-node clustering, availability groups with multiple and/or readable replicas and transparent data encryption. Oftentimes Standard Edition is a great way to save money, but it's important to be sure that the features you need are included and your uptime requirements can be met.
- Speaking of licensing, optimize for speed and efficiency with as few cores as possible to cut down on license costs. You need to test, evaluate and discuss with software vendors, but always remember that licenses are per core.

SQL SERVER CONFIGURATION BEST PRACTICES

AFTER INSTALLING WINDOWS

Storage

- Format drives for data and TempDB to 64KB allocation.
- Plan on having separate LUNs for data/log/system/TempDB/backup. Possibly separate large DBs to their own LUN. Even if they still go to the same device, this gives logical redundancy and can improve throughput in multipathing scenarios.
- Consider load testing the IO of your server before deploying to production. Diskspd is a good tool, or even Crystal Disk Mark for a quick sanity check.

Operating System

- Set CPU power saving mode to high performance (and set it in the BIOS level also, or the VMWare host level if on VM also). See more at Are Power Saving Settings Killing Your SQL Server Performance?
- Size the windows page file appropriately. With a dedicated SQL Server, you should not need a large paging file. The server dedicated to SQL Server should not be incurring much paging overhead.
- Consider verifying the patch level and performing necessary restarts before beginning SQL Server installation. Take advantage of the newness of the server and the lack of users, depending on uptime.

Security

- Set your anti-virus software to exclude data and log file directories. See more at How to choose antivirus software to run on computers that are running SQL Server.
- Restrict who has local administrative access to the server, as this can be a back door to SQL Server when given enough time and creativity.
- Provision an active directory service account. This service account does not need local administrative privileges. In fact, it is a security best practice not to have those permissions. The installer and SQL Server Configuration Manager will assign the necessary permissions to the service account. Also, never change a service account through the generic services control panel. Always do it through SQL Server Configuration Manager.

SELECTING A SQL SERVER VERSION

- Always consider the length of time before your next upgrade. It isn't unheard of to see people three or four versions behind in SQL Server. Unless a vendor gives a valid reason not to install latest version, that's exactly what you should install.
- Choose only the features you need. If you are not using SSIS, SSRS, or SSAS, there is no reason to select them, as they can always be added later. For larger and busier environments, those should probably be on their own server for performance reasons, though this will incur additional licensing requirements (but your users will thank you for the extra performance).
- It is considered a best practice to not install the client tools on your host. Starting SQL Server 2016, these are installed separately anyway. Microsoft has done this because the release cadence for the client tools is faster, and it is generally a worst practice to spend a lot of time remote desktop connected to the server itself. SSMS and other client tools can consume large amounts of memory and become a pressure point for SQL Server. You have to choose what is right for you but consider not putting the client tools on your server, but instead using a separate server with team access.
- Install binaries on D:\ (basically any drive other than C:\). Ideally, we should keep only OS and necessary components on C:\. At Straight Path, we prefer a separate drive for the SQL Server binaries – even separate from our data drives. This yields maximum flexibility and scale.
- Have SQL Server and SQL Agent Service start automatically.
- Disable SQL Server Browser unless using named instances.

- Set default drives for SQL Data and log files on separate drives from each other. Never install on C:\. Likewise, don't set backups to go to C:\.
- Install TempDB on its own drive. configured to guidelines in the following section.

CONFIGURATION BEST PRACTICES

There's still some work to do after SQL Server Installation, as many of the default settings are not optimal. Here are some considerations for optimal performance, keeping in mind that it's also a good idea to perform your own SQL Server assessment or have a third party review your setup.

Instance Level Configurations

- Set up SQL Server alerts, DBMail and operators. See the following for more information: How to Set Up SQL Server Alerts and Know if SQL Server is Healthy.
- Configure the max degree of parallelism (MaxDOP), which is generally one-fourth to one-half of the processors, to a maximum of 8, unless the installation is a data warehouse or has specific requirements otherwise. When in doubt, it's better to default to a lower value like 4 unless you are certain that your workload will benefit from a higher number. Leaving the server default of 0 only works on servers with 2 to 4 cores.

- Set the cost threshold of parallelism. We'll often start with a value of 50 here and then periodically review the plan cache and sometimes adjust down to 25 or higher. See more information at Tuning 'cost threshold for parallelism' from the Plan Cache.
- Determine the max memory settings. At Straight Path, we use a query based on this blog post to determine the right max memory settings: How much memory does my SQL Server actually need? (http://www.sqlskills.com/blogs/jonathan/how-much-memory-does-my-sql-server-actually-need/)
- Enable remote administrator connection.
- Cycle error logs at least weekly (daily is better).
    - Configure SQL Error Log rotation to 52 for a weekly rotation and 99 for a daily rotation.
- Enable trace flag 3226 to suppress backup successful messages in local install.
- Install SP_WhoIsActive (http://whoisactive.com/downloads/). Be sure to install it in the master database so it can be easily run from any user database.
- Set the backup compression configuration default to 1.
- Set optimize for ad hoc workloads to 1.

Database Level

- Configure the Model Database. Because it is the base for all new databases, any options not set on database creation in a create database script are inherited from Model.

        - Set to simple or full recovery model based on your backup approach. If you have a backup routine that backs up all user databases in full recovery model with a log backup, then full may make sense. If you want to explicitly use this recovery model for specified database, simple recovery model makes more sense.

        - Set log growth to a fixed value. The best value moving forward would be a bit higher, but some databases do not grow much, so start with 64MB on Model, and then evaluate the right sizing of the log file when making a new database case by case. 64MB should be fine for databases that do not grow and are just for containing scripts, tools and the like. See more information at Transaction Log VLFs – too many or too few?
        - Set the data file growth to a fixed value and not a percentage.
- Set up TempDB using the following information: Compilation of SQL Server TempDB IO Best Practices and follow these best practices:
    - For versions prior to SQL Server 2016, trace flags 1117 and 1118 should be enabled. See more at SQL 2016 – It Just Runs Faster: -T1117 and -T1118 changes for TEMPDB and user databases and TEMPDB – Files and Trace Flags and Updates, Oh My!.
    - Configure the number of data files based on the number of physical cores. General recommendations are one-quarter to one-half the number of CPUs, up to 8 files. Files need to be equal size with equal growth settings, set to a fixed value. Pre-grow to a potential maximum use to avoid needless file growths.
    - Pre-size the log file (there should be only one) appropriately to avoid auto grow events. A general starting point is at least 20 percent of the combined data file sizes. Keep auto growth to a fixed value and right size it from the start to avoid regular auto growth.
    - Monitor file sizes, percentage used, and auto grow events so startup file sizes can be adjusted to avoid auto grow events after a service restart.
- Follow these MSDB best practices:
    - Set recovery model to simple.
    - Pre-size data and log files to avoid auto grow events.
    - Set auto grow to fixed values.
    - Set up jobs to manage MSDB size, using the following commands:

- Sp_delete_backuphistory
- Sp_purge_jobhistory
- Sp_maintplan_delete_log

A NOTE ABOUT MAINTENANCE

Installing SQL Server correctly is only part of the equation of a well-run environment. SQL Server is not set-it-and-forget-it technology. It needs proper care and feeding throughout its lifecycle to bring the best performance, reliability and scale to your organization. This means after installation, you should perform regular monitoring, maintenance and even test restores of databases. Service packs and CUs should be evaluated and deployed after testing on a regular basis.

Maintenance, though, deserves a special call out. The scope of this document is focused on installation. But please pay attention to maintenance. At Straight Path we suggest using the Ola Hallengren maintenance solution (https://ola.hallengren.com/). While schedules and needs vary, you should regularly be performing these tasks on a schedule based on your needs, maintenance windows and user/business expectations:

- Index and statistics maintenance
- Regular backups (at least full backups, but usually log backups in line with your RPO and RTO, and sometimes differential backups to allow less frequent log backups)
- Error log recycles as described above
- Integrity checks looking for signs of database corruption
- Regular cleanup of maintenance, backup and agent job history to keep the MSDB database pruned and operational

STRAIGHT PATH SOLUTIONS

ABOUT US

OUR PLEDGE IS SIMPLE: DATA MATTERS. PEOPLE MATTER. RELATIONSHIPS MATTER, AND WE WILL DO RIGHT BY EACH WE ARE A PART OF.

At Straight Path, we know that strategically planning and managing databases can become a full-time job because, well, it is our full-time job. You shouldn't have to wade through the myriad issues that come with data management, such as licensing, upgrades, virtualization, performance and the like. You should be free to focus on your business. That's why we're here.

www.straightpathsql.com

Remote DBA

First-Class Remote SQL Server DBAs:
There when You Need Us

- Choose from flexible Remote DBA plans
- Expand your technology team
- Prevent problems before they occur
- Mentor your staff
- Enjoy 24/7 coverage

Health Checks

Do you know how to minimize risks to your client's data? Don't wait for a database failure to begin treatment. A SQL Server Health Check is a great place to start.

Have you given your SQL Servers a health check? Do you know what to look for?

Upgrades

Expert, Painless, Set-it-and-Forget-it Upgrades

- We plan the upgrade based on your needs
- Best practices applied
- Cooperative test run and walk-through
- We stay until things are good after go-live
- We leave you with peace of mind