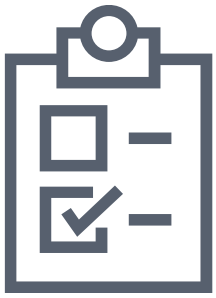
StraightPath Solutions

SQL SERVER UPGRADE CHECKLIST



Considerations Before, During, and After SQL Server Upgrades

Straight Path IT Solutions

www.straightpathsql.com

SQL SERVER UPGRADE CHECKLIST

TABLE OF CONTENTS

SQL SERVER UPGRADE CHECKLIST

INTRODUCTION

SQL Server upgrades don't have to be stressful! In 2019, Straight Path's founder, Mike Walsh, walked folks through the things to consider before, during, and after a SQL Server Upgrade process. You can watch the videos of this series over at our blog at the SQL Server Upgrades Made Easy page.

This document can help you get on the way to having a SQL Server Upgrade checklist. You'll need to cut and paste from this, add in the details about your own environment, test all the steps to be sure they work right for you, and use this as a guide for your own checklist creation process. But please, take all you can from this and use this as the baseline for your own checklist! If you have a good rollback plan, you may physically have a second chance to do your upgrade right! Don't burn the good will of your users and management by doing a SQL Server upgrade without planning!

This document won't answer your questions and do the steps for you! It would never end! It's to get you thinking of the questions to ask and steps to do! Look to our blog and videos for some thoughts on possible answers to the steps!

NOTE - As always – this is a document from the internet with ideas from our team and because you will be attempting these steps on your own and interpretations may vary, this is offered without warranty! Your mileage may vary and you should test these procedures and steps and make sure they work for YOU and your environment and make sure you are addressing all risks unique to your environment. This document is provided as a guide for you to create your own checklist! It comes from our knowledge and wisdom and the many successful upgrades we've done!

SQL SERVER UPGRADE CHECKLIST

BEFORE YOU EVEN START TO START

"The beginning is the most important part of the work."
- Plato, The Republic

A well-executed SQL Server Upgrade is best approached like a well-built home. You need to start with a solid foundation. Even before the foundation, though, you need to start with well-thought-out plans!

Every phase of a SQL Server Upgrade could be thought of as, "The most important" phase! And we sort of think that way here at Straight Path, too. But if you'll allow what might sound like a little hyperbole, the beginning really is the MOST important.

You need to answer the same questions your earliest teachers spoke about – Who, What, When, Where, and Why. (We'll cover How in later checklist steps!).

This stage is almost less about checklists of specific tasks or kinds of tasks, and much more about questions you should be asking.

The major goal for this phase – To identify the needs, know what you have, what you'll need and where you're going.

Before You Start to Start Questions

It's your job to answer these questions as you go down your upgrade path. Before you even start building the new and laying down the plan!

- ☐ **Why upgrade?** (Is management onboard? What features will you take advantage of? What is the "upside" to the bean counters and users?)
- ☐ **Which Version are we going to?**

- ☐ **What do we have for SQL features?** (Inventory your current environment, make a NEW checklist outlining: all features you have installed and are *actually using*, instance configuration, and settings.)

- ☐ **What Edition?** (If you are Enterprise today and pre-SQL Server 2014, you can use 128GB of RAM in Standard after you upgrade. If you are on any build prior to SQL Server 2016 SP1 today, you can use many Enterprise features in standard. Explore your options. Saving money could be a huge "Why!")

- ☐ **In Place or Migration?** (We hope you spend very little time here. For a list of reasons why the answer should be migration, check this DBA.Stackexchange answer out!)

- ☐ **Will the Cloud Be Involved?** (It's time to ask.)

- ☐ **Are we going to start High Availability or Disaster Recovery? Are we going to continue with the HA/DR we already are using?** (The options from SQL Server 2008 have changed! Some old standbys are there. There are many options. Maybe it's time to consider this finally! Especially if you can save license money moving to Standard!)

- ☐ **What will those software vendors say?** (You won't know if you don't involve them. Sadly, this is **not** optional! You **must** talk to your software vendors! Early. And sometimes, often. Ask questions like "What version do you support?", "Will going up break support?", "Do you use anything on the OS drive or registry that we need to account for?)

- [ ] **What do we have for needs OUTSIDE of the SQL instance on our server?** (You'd be surprised how many file paths are relied on locally on your windows server where SQL Server is installed. BCP paths? File shares? Filestream data? SSIS packages on the file system? DLLs for extended events or CLR?)

- [ ] **What is our upgrade team? (**Unless you are wearing ALL the hats of an IT team, you can't do your upgrade in "island mode" – who will represent the users? The vendors? The network? Security and Compliance? Management? Project Management? Dev? Systems? Storage?)

- [ ] **What kind of server should we move to?** (Without baselines and health checks, how will you know what resources you need? You can "guess" a little on a VM or Cloud VM – but you really can't when it comes to bare metal)

- [ ] **What help do you need?** (Are you good? Can you learn a few things from some online courses or free webinars like our webinar? Do you need some outside help to review plans and help bless it? You have one chance to get this one right. Review plans with team and make sure nothing is missing!)

- [ ] **What questions are missing for YOUR environment?**

Before You Start to Start Tasks

This is a good start list of tasks. The questions you come up with above may also create a few more tasks!

- [ ] **Inventory Your Environment** (DBAChecks.IO, Third Party SQL Documentation tool, etc.)

- [ ] **Perform a Health Check** (This can help document your current environment and teach you mistakes to avoid on the new! DBAChecks.io, firstresponderkit.org, dbatools.io, Glenn's scripts are all good free starts.)

☐ Baseline your SQL Server (What kind of performance do we need?)

SQL SERVER UPGRADE CHECKLIST

AFTER YOU DECIDE. BEFORE YOU UPGRADE.

Success is the result of perfection, hard work, learning from failure, loyalty, and persistence. - Colin Powell

Now you have an idea of why and where you are going. It's time to start transitioning to the "how" questions. There are still some of the "W" questions to ask here, though.

Let's Get Going Questions!

☐ What is the new SQL Server going to look like? (From above we should know the version, have a sense of the resource needs and know what the vendors will and won't allow.)

- ☐ **What mistakes from above will we fix?** (It's time to lay down a SQL Server **installation checklist** – don't worry, we have one of those, too, and it's free – go to our upgrade page and you'll see it in resources!)

- ☐ **Who is in charge?** (Someone must approve all this. Someone must set go/no-go standards and enforce them. Who is it? Do they agree? Half of our work is setting and managing expectations!)

- ☐ **We picked a version *we* want. Will our database agree?** (Yes. We need to test. You *could* go to a lower compatibility level and just ease some of that testing, but what a wasted opportunity!)

- ☐ **Who is going to sign off for the change?** (Let's assume you are going to test for that question right above this one! Who is signing off that we are good? Will they test thoroughly? We don't want problems on the Monday after upgrade weekend!)

- ☐ **Is the new server built with best practices?** (Do a health check on the new! Save it as a best practice config guide! Use our build checklist while you build. Use those health check resources. Engage with a consulting firm to give a blessing and leave you with a document and a "good housekeeping seal of approval!")

- ☐ **What about security?** (What kind of lapses have you had? What kind of mistakes did you see in the old and the health checks? Now is a chance to fix this. It's getting close to upgrade time, let's fix them! Think about new features like encrypted backups, TDE, etc.)

- ☐ **Are you documenting as you go?** (All the answers you are gathering about versions, needs, vendor support info, new server builds, SQL Server standards, SQL Server upgrade checklists. Steal our checklists and build your own to keep for next time and posterity!)

- ☐ **Choose how you'll handle connections and names?** (At Straight Path, we're huge fans of CNAMEs/Aliases – you'll have to deal with the pain one time of using them, but if you start using names not related to the server moving forward, your next migration upgrade will be a bit easier.

No more hunting for connection strings in apps and changing them! – We talked a bit about this at length in the week 3 video on our upgrade page)

- ☐ What will upgrade week look like? What about cutover night? (Start documenting that! Use our "During the upgrade" checklist in the next section to get a start!)

- ☐ What is our window? (What uptime requirements do you have? How big are your databases? Regardless of the size of the DB, you can support a super-low cutover *if you need to* with full/diff/log backup restore approach described below. Figure the window out ahead of time and get that team we talked about two sections above to agree!)

- ☐ Are our vendor support people there for us? (We need them now! Will they be there for go live night? Before? Do we have a blessing e-mail from them?)

Let's Get Going Tasks!

- ☐ Run Upgrade Advisor. (It's about due diligence. Run it.)

- ☐ Consider looking for deprecated features. (You can use Extended Events, you can run a build in SQL Server Database Tools)

- ☐ Obtain tests and signoff. (The more time you spend here, the less time you spend on go live weekend or supporting problems on the first live day!!)

- ☐ Build your new home with a plan. (*Use our free SQL Server build document available from the upgrade page*. Add your own standards, build with success in mind! Remember ALL the features and configs you looked for above, in fact if you have more than a plain SQL Server environment, you really ought to create a separate checklist just with the features and parts and configs you need on the new before you even build your new SQL Server!)

- [ ] **Do an initial health check on the new server.** (Save it, this can be a runbook/config guide of sorts.)

- [ ] **Create some documents and checklists – use ours as a start!** (Us technologists don't love them – until we need them! Build some docs about the plan, build your migration checklist based on the below sections, document your migration plan, document your rollback plan, get agreement up front on what "success" is.)

- [ ] **Practice your migration! (**If you are following our advice and doing a migration upgrade instead of an in place, what better way to test the steps and the new environment than testing ON THE NEW ENVIRONMENT? Rehearse your steps. Pretend something went wrong. Rollback. Understand what you are about to do!)

- [ ] **Tabletop talk-through** (This may sound like paranoia – maybe it is, it's a good trait for a DBA, but start talking about what *could* go wrong. Invent scenarios of problems and discuss fault lines that exist in the various disciplines. As you discuss these things and "imagine" your upgrade and what could break, you'll come up with new checklist items and you'll come up with ideas of things you're forgetting about!)

- [ ] **Go Learn about DBATOols.IO!!!** (PowerShell scripts to make migrating EVERYTHING easier!! www.dbatools.io – free and awesome! You can even do your full migration with their Migration commands, but we typically use all the various copy-DBA* commands to bring over creds, jobs, linked servers, etc.)

- [ ] **Agree with how you'll confirm you have all the data and how you will kick users out.** (Be it looking at running a record count query before and after, looking at the most recent record in an often-used table, be it a policy of "We told them, if someone was doing something in the window that's their problem!" – we don't love that one, by the way…., or whatever method you have in mind – figure that out ahead of time. Also agree how you will shut down access to SQL at cutover ahead of time! Kill inbound firewalls, put up maintenance pages, shut down apps, etc.)

SQL SERVER UPGRADE CHECKLIST

UPGRADE WEEK TO GO-LIVE

"There's 106 miles to Chicago, we've got a full tank of gas, half a pack of cigarettes, it's dark out, and we're wearing sunglasses."
"Hit it!"
-The Blues Brothers

This is the reason why you're here. Let's go get upgraded. We'll dispense with the questions here and give an idea of a possible go live checklist for the week of go-live. Because we don't recommend an in-place upgrade, we're assuming a migration upgrade. We're also going to assume you want a fast cutover, it's easier to build a checklist with a slower cutover.

NOTE: We do not like or advise you to do a detach/attach method. For the same reason we don't love cut and paste. Detaching a database removes SQL Server's file handle on it. Once you've been around for the mangling of a detached DB without a recent backup or a big DB with a slow restore, you'll appreciate this advice! We have those scares so you don't have to!

We'll make a fictitious "upgrade week" here. "T" = the time of cutover and we'll organize the steps walking forward to upgrade. We're assuming you've already built a new server using our build guide combined with your own, we're assuming you have sign off and you have a rollback plan, we're assuming you

have a support team ready, we're assuming you're good to go and all testing is great!

T Minus 1 Week

We're a week before the upgrade. What should we be doing?

- ☐ Obtain final sign off (From all teams and disciplines? Are we go? Think NASA launch control. "We are going to start upgrade steps. At the conclusion of this week at such and such a time, we'll be cutting over. Is each team good?" Have that conference call and obtain sign off from all.)

- ☐ CODE FREEZE / CHANGE FREEZE (Not *always* possible but should be possible more often than many allow for! Minimize changes to the old environment. The less change, the less chance for risk!)

- ☐ Put new environment into a clean state (Wipe out what needs to be wiped out – though see next thought as well – from testing, clean up anything that doesn't need to stay for production.)

- ☐ Depending on how well you locked down changes, consider moving items that can move ahead (Linked severs, logins, jobs, certificates, credentials, etc. If you KNOW it won't change and you know you won't need it on migration night, bring it over – and find your checklist for all the items you want to bring over and check those off. Celebrate the early start of checking migration items off!)

- ☐ Confirm phone tree/setup go live meeting (Your full support team, management, application owners, vendor support personnel, line them up and either have them on a call or available where they can be reached)

- ☐ Agree on Go Live Night "Acceptance" criteria (Who will smoke test? What will they check? What black and white criteria constitutes a success vs a rollback?)

T Minus "A few" Days or hours

We are some number of days before cutover. We'll say 2 days before cutover night in our imagined world here. This also assumes your databases are in FULL recovery model. If your databases are very small and could be backed up, copied, and restored in 5-15 minutes, this could be omitted for those DBs and you could just do a full backup at cutover time! But for larger DBs it may be worth looking to full recovery model and utilizing log backups to minimize downtime at cutover, if needed.

- ☐ Confirm everything (No one ever got in trouble for triple checking. Approvals still good? Any code changes? Anything missed? How's the checklist looking?

- ☐ Take a full backup of all DBs migrating and start copying them to the new server (We are going to restore them in a moment. With NORECOVERY. You are playing with your recovery chain a bit here if you take Diffs or logs. You should take a REAL FULL backup – not a copy only, we'll take a diff the day of cutover, or day before.)

- ☐ Adjust your backup schedule (We are going to take a differential before cutover. Either a day before and then go to log backups or a few hours before and then go to log backups. Make sure you don't get another full backup in between there. But make sure you DO get regular full backups. Just because you are migrating this weekend it doesn't mean you are immune from a terrible coincidence! Always be prepared to recover!)

- ☐ Restore those full DBs to the new server WITH NORECOVERY (we are doing a transaction log style migration. Now if you have a lot of DBs you should modify these steps and consider maybe actually using log shipping,etc.)

T Minus Hours through T-Minus Seconds

It may be days if you have big databases. You can script transaction log restores so it isn't hard to have to copy many and do many restores. But we would rather be lazy here and wait until the last minute that we can safely take differential backups, copy them, restore them and have as little time of log backups to restore. And depending on how busy an environment is we may even change the T-Log backup frequency from 5-15minutes to every hour after this diff. It's fewer files to copy over!

- ☐ Since the differential backup was taken, you should be copying your log backups to the new server and restoring them WITH NORECOVERY (Again, this is basically you doing log shipping. Start copying your log files over and restoring. This is why we like taking the diff at the last possible minute. But if you have VLDBs and need to copy sooner, you can build jobs to accomplish these steps or just use log shipping.

- ☐ Continue restoring the logs until you get to the Cutover window. Always specifying WITH NORECOVERY.

- ☐ Make sure no DB modifying jobs are due to kick off right at your cutover window. (it's embarrassing to have a miss here! It's the one thing you, as a DBA, have full control over!)

Cutover Window

We are about to pull the trigger! The steps here really can vary. Even when we help a client it depends on many factors. The main goals here are

- Keep users out and make sure no data is entered after final log backup and restore is done.
- Kick everyone out and do due diligence
- Make sure old DBs go offline ASAP so no false positives happen connecting to old

- The final log backups are taken and brought over
- We've verified we're good.

The Steps here could look something like:

- ☐ Someone confirms "We've shut everything down" (Someone other than the DBA! The app teams/the firewall teams/the PM – you are at cutover window and no one/no app/no service should be connecting. They said "we're down")

- ☐ Be paranoid and make sure! (look for activity, consider bouncing SQL Server service just in case something was still connected. If they are changing CNAMEs or pointers, consider moving those at that time also – due diligence.)

- ☐ Quickly take your final Log backup and then obtain your row counts/run your verification query and OFFLINE the Databases (Be ready to go once you bounce SQL service, do the log backup and THEN the counts and OFFLINE the databases. This way you are doing the most you can to be sure you have the row counts last. If something happened immediately after your log backup but before your counts – you'll have a mismatch and you can sort out how to deal with it! Offline the DBs but not the server so you can still reference settings if needed during testing, also a great tool to find connections still trying to hit the old server – more on this later)

- ☐ Restore the final log backups on the new server WITH RECOVERY (moment of truth! We are applying the final transaction log backup and running recovery. This is the actual cutover and the time should just be these steps in this section on this page. This is why we do this approach)

- ☐ Run your row count queries (does everything match? If not – figure out why and deal with it.)

- ☐ Smoke Test (That's the team you should have identified at least a week before tonight)

- [ ] Rollback or Open Access (If testing went well, start letting real people in. If it didn't, start rolling back!)

- [ ] Monitor New Server. (The next moments and hours and days are important!)

- [ ] Monitor Old Server. (I love leaving the old server online with all DBs offline. Look at your SQL Server error log. See any failed logins because DBs are not available? Hunt down those IPs and apps and figure out why they are still trying to connect to the old server!)

Afterwards

Once you are past a couple days or so of live on the new, you can pat yourself on the back. You made it! The old server can go away once you are satisfied nothing is trying to connect to it. You should have this new server being monitored from cutover night with your tools. Now it's time to do the DBA thing to your new server!

CONGRATULATIONS!

SQL SERVER UPGRADE CHECKLIST

ROLLBACK!

A quick note on rollback. You should make a rollback checklist. You should assume the worst case. If you are doing a migration-based upgrade the rollback is as simple as pointing connections back to the old and bringing those DBs back online.

If you are doing an in-place upgrade, your steps will vary from lucked out

simple to “GET HELP!!!!!”. Really, we strongly dislike in-place upgrades. Even with a migration-based upgrade, though, you should create a rollback plan. You should document who to call and what to do and you should have mitigation plans.

You should also identify questions like “How long will we troubleshoot before giving up? Who will make the call?” well in advance. With a migration upgrade, there’s always a simpler roll back plan and there’s always next week! Once you get into troubleshooting for more than a few hours or a whole overnight it gets ugly fast and you start making mistakes. Rolling back is a sign of intelligence and strong DBA skills, not a sign of weakness! Not rolling back when you should have is a sign you are suffering from sunk cost fallacy.

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