

CHECKLIST FOR SQL SERVER 2014 UPGRADE

Ensuring the Best Outcome for Your Upgrade

By Thomas LaRock, Technical Evangelist and Head Geek



INTRODUCTION

Sometimes, database administrators may make the mistake of believing that the upgrade process for a database server is as easy as pressing a few buttons. The truth, often arrived at through painful experience, is that an upgrade process can be very complex. A proper upgrade process involves detailed research, planning and execution. This document will help you to better plan and execute a successful upgrade to SQL Server 2014.

HOW TO USE THIS CHECKLIST

To help you get started, I've compiled this list of pre- and post-installation tasks to include in any checklist you put together for migrating your database server to SQL Server 2014. Including these items into your checklist is likely to help you avoid most, if not all, potential upgrade issues.

Please note that these steps are specific for an upgrade to the database schema and data. They do not include anything regarding the upgrading or testing of an application that is going to be accessing the upgraded database. You will want to remember to test your application and not just assume it will work perfectly even after the database has been upgraded. I would also advise that you perform these steps in a non-production environment first because I often find that common sense isn't so common after all.

ABOUT SQL SERVER 2014

SQL Server 2014 represents a significant advancement in a true hybrid database platform, and offers a host of exciting new features, including:

- » A new **in-memory OLTP engine** (often referred to as Hekaton)
- » **Deeper integration with Windows Azure**
- » **Memory caching using SSDs**
- » **Updateable tables that are using Columnstore indexes**
- » **A Resource Governor that can throttle I/O**
- » **Enhanced cardinality for statistics**
- » Enhanced separation of duties
- » Expanded the ability to do ONLINE maintenance operations

Some of these will come with new hardware requirements. Some will require configuration changes. It is critical to become familiar with the new features and understand how they will impact your environment.



A note about migrations vs. in-place upgrades

This checklist is meant to serve as a guide for either an in-place upgrade or migrating from a previous version. However, I always recommend doing a migration whenever possible instead of in-place upgrades. Migrations (typically done by restoring a database backup from the current instance) just make me feel more comfortable should I need to troubleshoot issues later. You should do what works best for you.

START WITH THE SQL SERVER 2014 UPGRADE ADVISOR

The SQL Server 2014 Upgrade Advisor (SSUA) is an advisor. Much like a consultant, it doesn't fix everything that is wrong. Instead, it advises you on what actions you should take when upgrading to SQL 2014. The actions the UA recommends will come in two forms:

- » Recommended actions to be done prior to a migration
- » Recommended actions to complete post-migration

While the UA is very good at finding the things that need fixing prior to a migration, it isn't fool-proof. You should do your homework thoroughly before migrating to a new version of SQL Server. This checklist will help you do just that.

REVIEW THE "BREAKING CHANGES" SECTION IN MICROSOFT'S BOOKS ONLINE

Microsoft publishes a list of breaking changes for each version of SQL Server in its Books Online (BOL) section, and you should review these lists prior to any upgrade. By doing so you will be able to better address the occasional odd behavior you might encounter in your SQL Server environment during post-migration testing. Microsoft's BOL is more dynamic than the SSUA and may have an entry or two that doesn't make it into the UA checklist.

REVIEW THE "BEHAVIORAL CHANGES" SECTION IN MICROSOFT'S BOOKS ONLINE

Similar to the breaking changes, the behavioral changes are worth reviewing. One reason the behavioral change list is so important is that it includes things that the SSUA is likely to never report because the behavioral changes list refers to functionality that could break, not just things that will break. Also worth noting is that the BOL appears to have two entries: one for SQL Server features, and one specific for the database engine.

EXECUTE DBCC CHECKDB WITH DATA_PURITY

One of the post-migration or upgrade tasks you should be sure to complete is to run the following statement:

```
DBCC CHECKDB WITH DATA_PURITY;
```

This statement will check for values that are no longer valid for the column datatype. For databases created prior to SQL Server 2005, this step is very important. For databases created in SQL Server 2005 and later, the DATA_PURITY check should be done automatically with a regular CHECKDB.

NOTE: If you have a database that was created in SQL Server 2000 that has been migrated to a SQL Server 2008 instance, and left in the SQL Server 2000 (80) backward compatibility mode, you should not assume that the DATA_PURITY check has been getting done. Run it yourself now to be certain it is getting done.

EXECUTE DBCC UPDATEUSAGE COMMAND

While not as critical as the DATA_PURITY command noted above, you should run the following command for any migration or upgrade process:

```
DBCC UPDATEUSAGE(db_name);
```

This **command** will help fix any page count inaccuracies that are resulting in the sp_spaceused stored procedure returning wrong results.

For SQL Server 2012, this check was recommended for databases created prior to SQL Server 2005.

For SQL Server 2014, the BOL entry link lists this command as being applicable for databases created in SQL Server 2008 and later. That seems odd since this command is valid for SQL Server 2005. I'd recommend you run this for SQL Server 2005 databases being migrated to SQL Server 2014 anyway.

IMPORTANT: ALWAYS UPDATE STATISTICS!

For any migration or update checklist, it is critical to run the following:

```
USE db_name;  
GO  
EXEC sp_updatestats;
```

This command will update the statistics for all the tables in your database. It issues the **UPDATE STATISTICS command**, which warrants mentioning because you may want to use that command with the FULLSCAN option.

I'm the type of person who would rather be safe than sorry and would rather end up running something like this:

```
USE db_name;  
GO  
EXEC sp_MSforeachtable @command1='UPDATE STATISTICS ? WITH FULLSCAN';
```

Bottom line here: don't forget to update the statistics. Failure to do so could result in your queries running slowly as you start your testing. If this happens, you may end up wasting your time while you try to troubleshoot the possible bottlenecks.

With SQL Server 2014 there is also a new Cardinality Estimator (CE), and I'll talk more about this later, but you are going to want to make certain your statistics are as accurate as possible before you begin any testing. So, take care of the statistics now, and you don't have to worry about it later.

REFRESH VIEWS USING SP_REFRESHVIEW

Believe it or not, every now and then someone will build a view that spans into another database on the same instance. Sometimes these views will go across a linked server as well. Your view may not be contained in just your database. In what could be the most dramatic twist of all, sometimes these views are created using a SELECT * syntax.

What are the odds that you could have such code in your shop? It happens. And when you have bad code on top of views that go to other databases (or views of views of views of whatever else some sadistic person built) you are going to want to use [sp_refreshview](#) to refresh those views.

So, if you are migrating a database in your environment to a new server, you should refresh your views using [sp_refreshview](#). Most of the time, this won't do anything for you. But there is that one chance that it will dramatically improve performance.

FORGETTING TO TAKE BACKUPS

You should always make a backup prior to the start of any migration, and you had better take one right before you turn that database over to your end users. Also, you should save any output from the seven items listed above, as it could prove helpful should something go awry later. (Bonus tip: [make sure your backups are good!](#))

UPGRADE YOUR HARDWARE, IF NEEDED

Be sure to review the [minimum requirements for installing SQL Server 2014](#). Be aware that the requirements listed are the minimum requirements. If you are upgrading, then it might be time to upgrade your hardware as well, and maybe consider virtualization.

The real reason to consider upgrading your hardware is to take full advantage of SQL Server 2014 features. Considering [the number of shiny new things in SQL Server 2014](#), you will likely need to do some extra legwork to determine what hardware you'll need. For example, Hekaton, the true in-memory OLTP (on-line transaction processing) engine that is the most talked about feature of SQL Server 2014, isn't specifically addressed in the minimum requirements, but [this blog post from the SQL Server team](#) suggests some guidelines.

KNOW THE RIGHT MIGRATION PATH

For those running SQL Server 2000 instances, you can't upgrade directly to SQL Server 2014 without first upgrading to an intermediary version. In this case, you have two options when going from pre-SQL Server 2005 versions:

- » You can do an upgrade in place to SQL Server 2005, SQL Server 2008, SQL Server 2008 R2, or SQL Server 2012
- » You can do a backup (or even detach) your database and restore/attach to an instance running SQL Server 2005, SQL Server 2008, SQL Server 2008 R2, or SQL Server 2012. At that point you will be able to complete the upgrade to SQL 2014.

As noted earlier, I prefer to do migrations rather than upgrades in place. For something like a SQL Server 2000 database, I would do a backup, restore it to an intermediary version (like SQL Server 2008 R2), then do another backup and restore that to SQL Server 2014.

CHECK COMPATIBILITY LEVELS

If you've done SQL Server upgrades before, you may have noticed that the compatibility level does not get set to the newest version after the migration is complete. You must manually **set the compatibility level yourself**.

This is especially important with SQL Server 2014 because of the new Cardinality Estimator (CE). In most cases, the new (CE) will have a positive impact on performance, but in some circumstances it may actually harm performance. It's important to first understand how, why and when SQL Server 2014 uses statistics, and then when and where the new CE may impact for your performance. Refer to my [TechEd presentation](#) or this [whitepaper from Joe Sack](#) to learn more about the new CE.

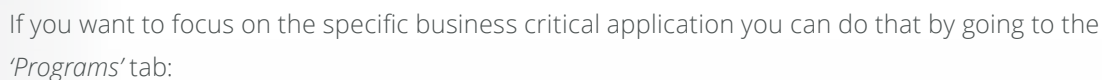
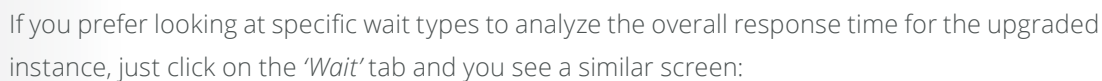
READ THE RELEASE NOTES

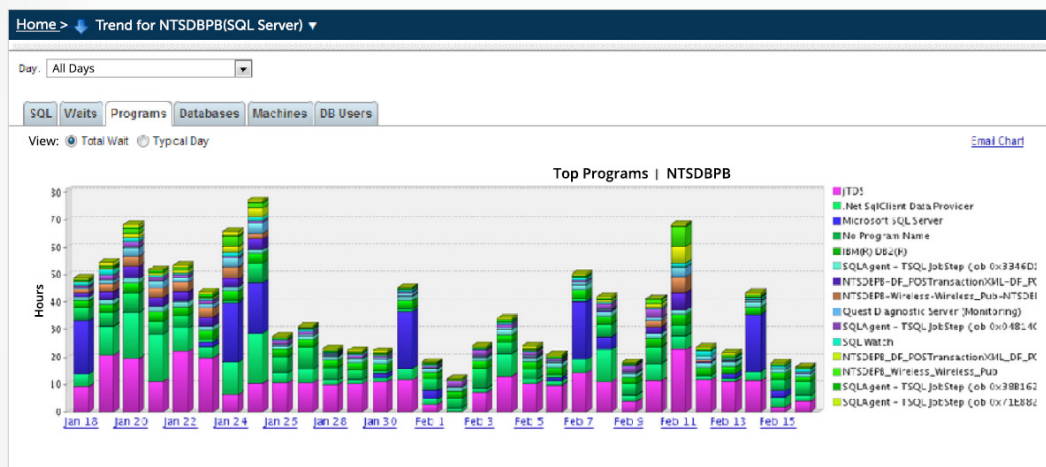
Take a few minutes and read the release notes. It's good to have as complete a picture as possible for the new version in case something does not work as expected, and there are details in the release notes you may not find elsewhere.

FINALLY, ESTABLISH A BASELINE FOR NORMAL DATABASE WORKLOAD, RESOURCE USAGE AND RESPONSE TIMES BEFORE THE UPGRADE

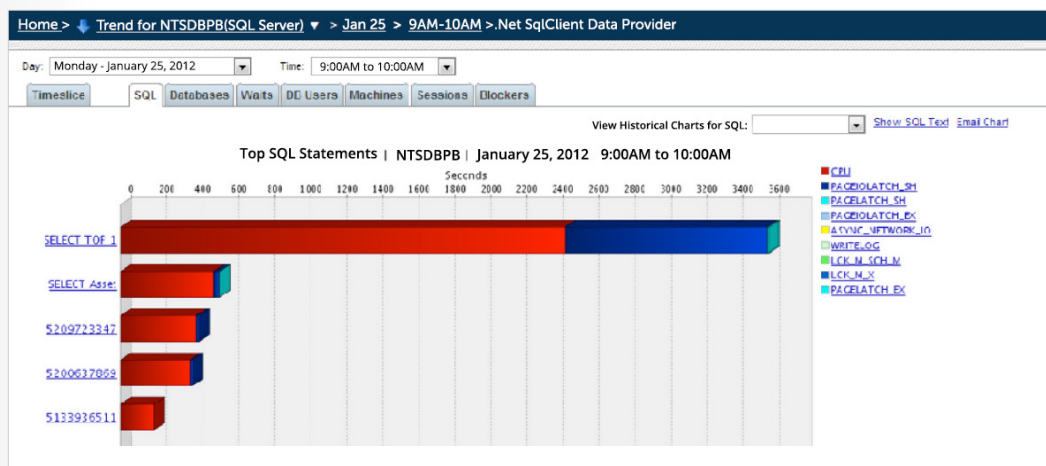
If you have an initial baseline for normal workload resource usage and response times prior to an upgrade, then you can objectively assess performance before and after, and you can proactively assess any problems introduced by the upgrade. In fact, this is critical to do for any change event. Tools like the [Distributed Replay Utility](#) allow for you to capture a live workload that can be replayed against your upgraded test server. Running a continuous database performance monitoring tool such as SolarWinds Database Performance Analyzer (DPA) against both servers will allow for you to easily compare and contrast performance for your key business critical systems.

Want to see the overall trend analysis for all the SQL statements running against your new server? That's easy, simply go to the home screen for the database instance:

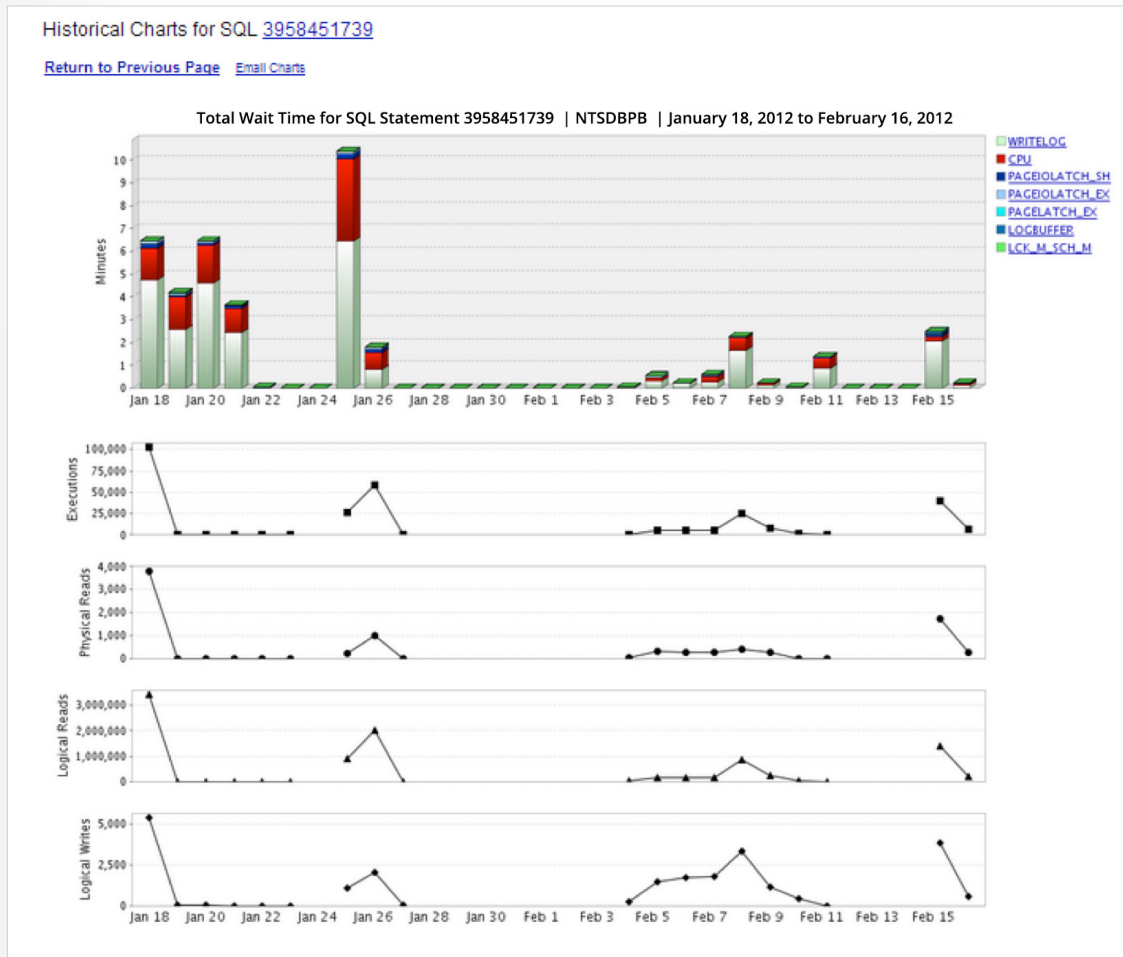




You can then drill into a specific day, then hour, and then select the program name. You will then see a list of specific SQL statements that were executed by that program:



In the upper right there is the ability to quickly view the historical chart for any SQL statement:



At this point you can compare these results to the results on the original production server and verify if performance is acceptable before completing the upgrade process by switching your database connections to point to the new server.

IT'S ALL ABOUT THE PREPARATION

Upgrades are a necessary part of any development lifecycle. The chances of having a successful upgrade increases along with the amount of planning and preparation you invest in building a proper upgrade process. If you are planning to upgrade to SQL 2014, you can use this paper as a guide to developing your own checklist.

Incorporating DPA into your upgrade process will allow for all stakeholders in the project to easily interpret database performance metrics at every step. Offering insight to database performance during the upgrade process will reduce surprises and help ensure a smooth cutover when you transition to the new server.

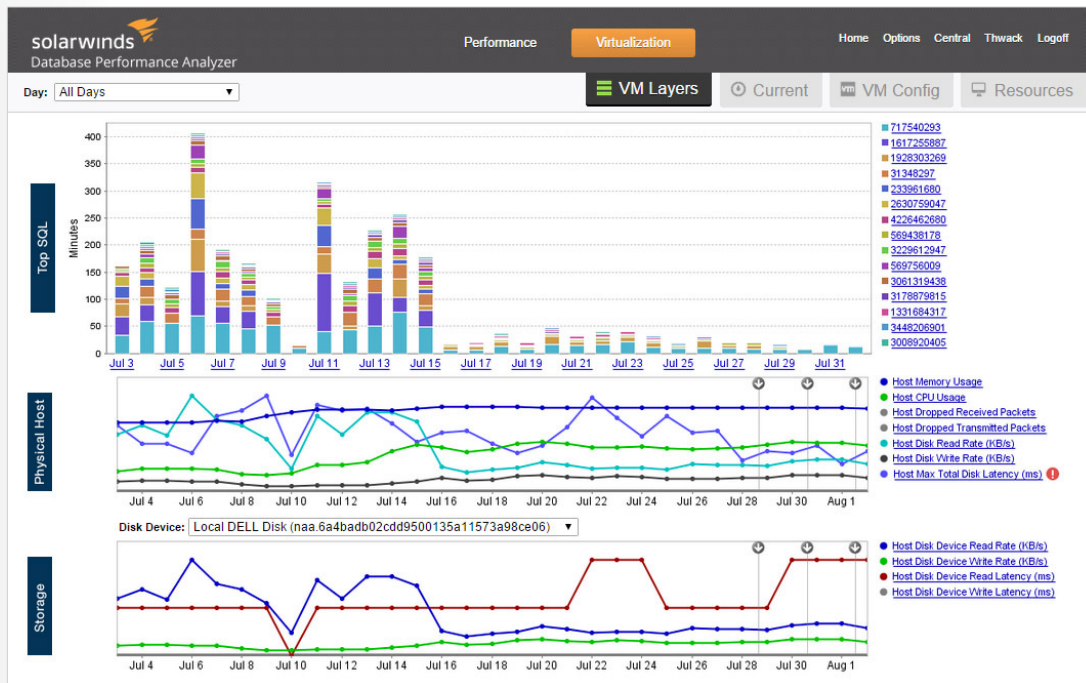


HOW CAN DATABASE PERFORMANCE ANALYZER HELP?

Database Performance Analyzer (DPA) from SolarWinds (NYSE: SWI) provides the fastest way to identify and resolve database performance issues. DPA is part of the SolarWinds family of powerful and affordable IT solutions that eliminate the complexity in IT management software. DPA's unique Multi-dimensional Database Performance Analysis enables you to quickly get to the root of database problems that impact application performance with continuous monitoring of SQL Server, Oracle, SAP ASE and DB2 databases on physical, Cloud-based and VMware servers.

[LEARN MORE](#)
[DOWNLOAD FREE TRIAL](#)

Fully Functional For 14 Days



For additional information, please contact SolarWinds at 866.530.8100 or e-mail sales@solarwinds.com.

