Modernizing your SQL Server the right way

Slides in http://aka.ms/tigertoolbox

Under "Sessions" folder

Microsoft

PEDRO LOPES

Sr. Program Manager, SQL Server Engineering

/pedroazevedolopes

@SQLPedro

Focus Areas

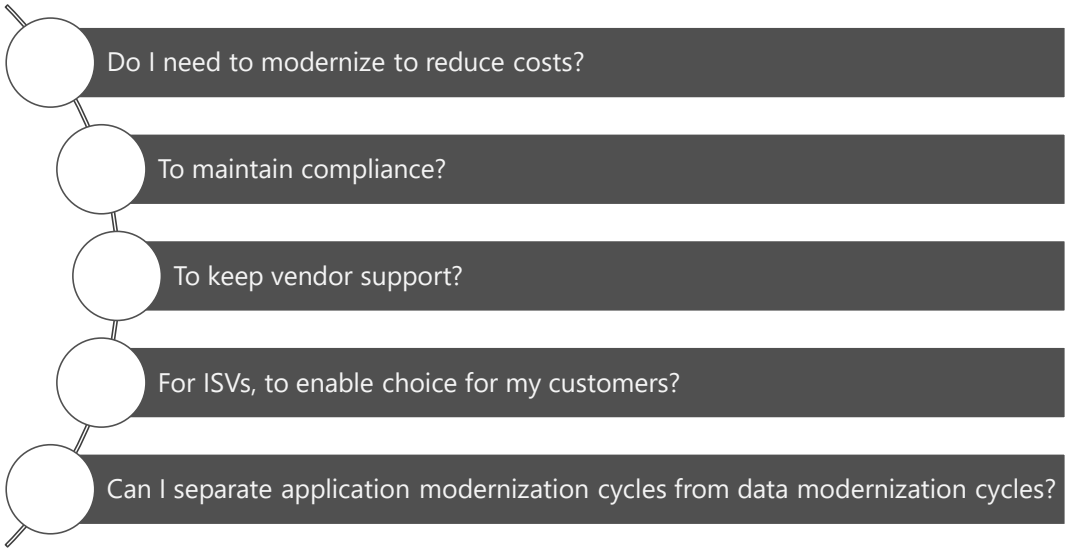
Relational Engine: Query Processor, Programmability, Performance

Agenda

- Why Upgrade?
- Database Compatibility Certification
- Upgrade Process
- Post-migration

Why upgrade?

Is it the right time to modernize?



As you may have observed, we released SQL 2016 about 24m after 2014, and 2017 about 16m after 2016. So there is a faster engineering cycle that's observable to meet market demands and pace of technology improvements. It really doesn't make sense that our customers would have to wait 2+ years to see innovations coming to their non-azure Data solutions, like it had happened historically.

Users and ISVs all face a fundamental and recurring question with their application and database estates: Is it the right time to modernize?
With that question come a number of considerations:

- Do I need to modernize to reduce costs? To maintain required industry certification? To keep vendor support? For ISVs, to enable choice for my customers?
- And how to minimize risk? Can I separate application modernization cycles from data modernization cycles?
- And what's my required certification process? What's the cost of re-certifying for Azure SQL, or newer SQL Server version?

Winter is coming…

July 9, 2019 – End of Support for SQL Server 2008 and 2008 R2
January 14, 2020 – End of Support for Windows Server 2008 and 2008 R2

https://www.microsoft.com/en-US/sql-server/sql-server-2008

Database Compatibility Certification

Database Compatibility Level based certification

Stop certifying for any given platform (Cloud, on-prem)!

Stop certifying for a named SQL Server version!

Any certification process should be thought in terms of "*which target database compatibility level am I certifying to?*"

We believe this paradigm shift unlocks agile modernization cycles and reduces upgrade risks.
And we are backing it up with actual engine level protections, as well as updated public documentation that moves in this direction: see http://aka.ms/dbcompat.

Key Benefits

- Simplified application certification on-premise and Azure (e.g. Azure SQL DB MI)
- Ability to provide customer a choice of latest SQL Server platform based on certified DB compat level
- Improved risk management by decoupling application upgrade cycles from Database upgrade cycles

Microsoft stands by DB Compat based certification

Microsoft Database Compatibility Level Protection

Full Functional protection once assessment tools runs clean with no errors

Query Plan shape protection on comparable hardware

Maintaining backward compatibility is very important to SQL Server team

Database Compatibility Level behavior

Database Compatibility Level sets certain database behaviors to be compatible with the specified version of SQL Server.

- Compatibility level affects behaviors only for the specified database, not for the entire server.

Product	Compatibility Level Designation	Supported Compatibility Level Values
SQL Server 2019	150	150, 140, 130, 120, 110, 100
SQL Server 2017	140	140, 130, 120, 110, 100
Azure SQL Database	130	140, 130, 120, 110, 100
SQL Server 2016	130	130, 120, 110, 100
SQL Server 2014	120	120, 110, 100
SQL Server 2012	110	110, 100, 90
SQL Server 2008 R2	100	100, 90, 80
SQL Server 2008	100	100, 90, 80
SQL Server 2005	90	90, 80
SQL Server 2000	80	80

New Databases are set to compatibility level mapping to the version of the Database Engine, unless the **model** database has a lower compatibility level.
When a database is upgraded from any earlier version of SQL Server, the database **retains its existing compatibility level** if it is at least minimum allowed for that instance of SQL Server.
Upgrading a database with a compatibility level lower than the allowed level, sets the database to the lowest compatibility level allowed.
Azure SQL Database V12 was released in December 2014. One aspect of that release was that newly created databases had their compatibility level set to 120. In 2015 SQL Database began support for level 130, although the default remained 120. Starting in **mid-June 2016**, in SQL Database, the default compatibility level are 130 instead of 120 for **newly created** databases. Existing databases created before mid-June 2016 are not affected, and maintain their current compatibility level (100, 110, or 120).
If you want level 130 for your database generally, but you have reason to prefer the level 110 **cardinality estimation** algorithm, see ALTER DATABASE SCOPED CONFIGURATION (Transact-SQL), and in particular its keyword LEGACY_CARDINALITY_ESTIMATION = ON.
For details about how to assess the performance differences of your most important queries, between two compatibility levels on SQL Database, see Improved Query Performance with Compatibility Level 130 in Azure SQL Database.

Functional change protection – Clarifying the caveats

- Breaking Changes = behavior changes resulting in different outcome
- Protected by Database Compatibility:

```
DECLARE @value datetime = '1900-01-01 00:00:00.003'
SELECT CAST(@value AS datetime2)
```

  - In DB Compat 120 or lower, result is:
    `1900-01-01 00:00:00.0030000`
  - Under DB Compat 130, these show improved accuracy by accounting for the fractional milliseconds, resulting in:
    `1900-01-01 00:00:00.0033333`
- Not Protected by Database Compatibility:
  - The query below works until DB Compat 90, but errors out starting with Database Compatibility 100 (error 241, conversion fail):
    `SELECT DATEPART (year, '2007/05-30')`
  - Instead use:
    `SELECT DATEPART (year, '2007/05/30')` or `SELECT DATEPART (year, '2007-05-30')`

Until DB Compat 90, datetime intrinsics such as DATEPART do not require string input values to be valid datetime literals. For example, SELECT DATEPART (year, '2007/05-30') compiles successfully.

Starting with DB Compat 100, datetime intrinsics such as DATEPART require string input values to be valid datetime literals. Error 241 is returned when an invalid datetime literal is used.

Functional change protection - Clarifying the caveats

- Deprecated = avoid use in new development
  - Deprecated functionality introduced in a given SQL Server version is still protected by that compatibility level.
- Discontinued = removed from product
  - Discontinued functionality introduced in a given SQL Server version is not protected by compatibility level.
- Removed T-SQL syntax.
  - In SQL Server 2012 the `fastfirstrow` hint was removed.
  - Regardless of the compatibility level, the query below will produce error 321 (not a recognized table hints option):
    ```
    SELECT * FROM HumanResources.Employee WITH (FASTFIRSTROW);
    ```
  - Instead use:
    ```
    SELECT * FROM HumanResources.Employee OPTION (FAST = <n>);
    ```

Feature Roadmaps

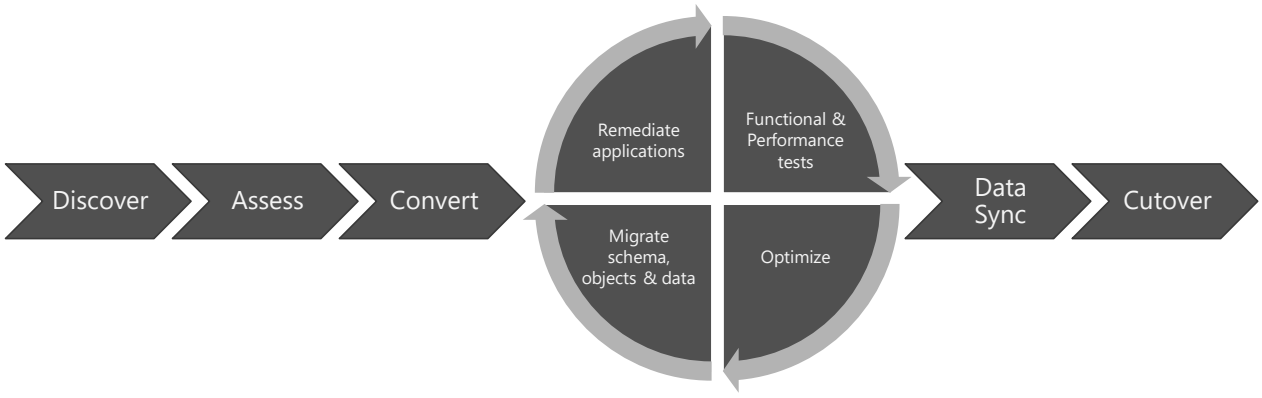
- For new development work, or when an existing application requires use of **new features**, as well as **performance improvements** done in the query optimizer space, plan to certify on the latest database compatibility level…

Upgrade Process

Traditional Upgrade Strategies

Feature	Notes
Log Shipping	Cutover measured in (typically) minutes
Replication	Cutover measured in (potentially) seconds
Backup and Restore	This is going to take a while!
Filesystem/SAN Copy	Ditto - the latter being significantly faster
Availability Groups (NOT available in < 2012)	Cutover measured in (typically) seconds

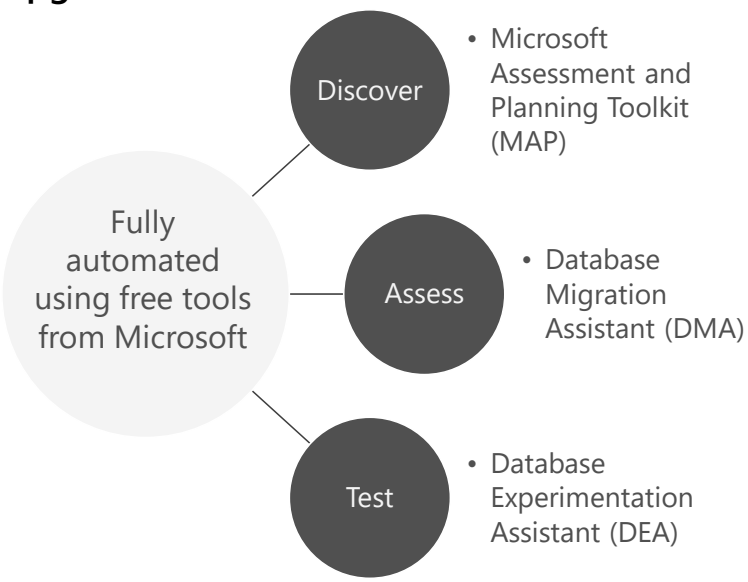
Minimize Risk with the Database Migration Guide



Migrations should be approached with the same rigor and processes as a full software or hardware project – a solid methodology is required for success. Microsoft provides you with all the tools you need to achieve a seamless, reliable upgrade experience.

The Database Migration Guide provides step-by-step guidance on reliable upgrade methodology.
https://datamigration.microsoft.com/scenario/sql-to-sqlserver

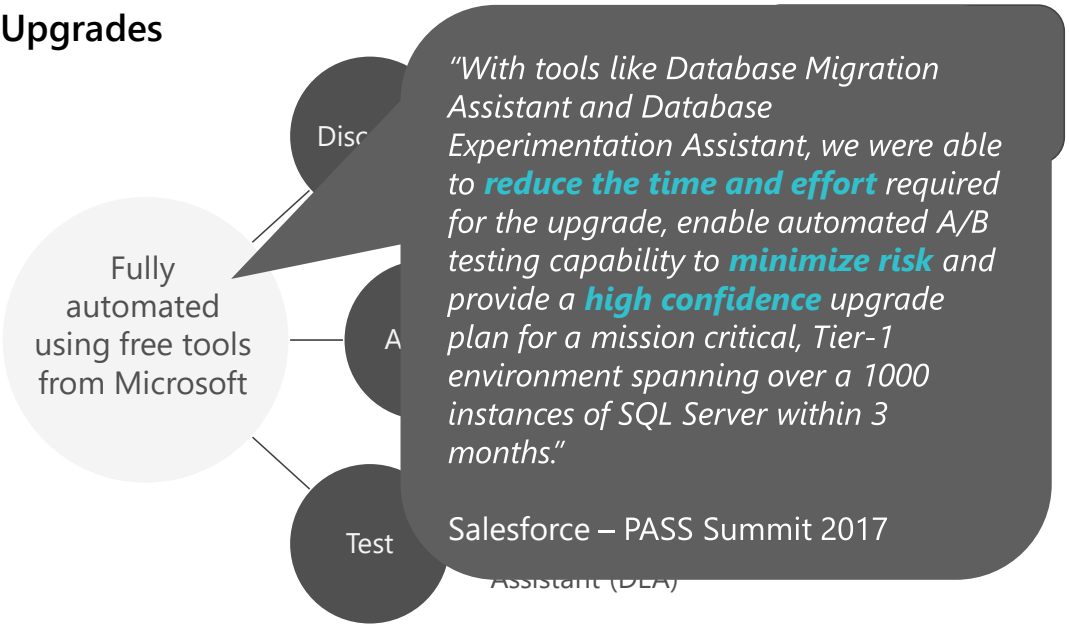
Reliable Upgrades



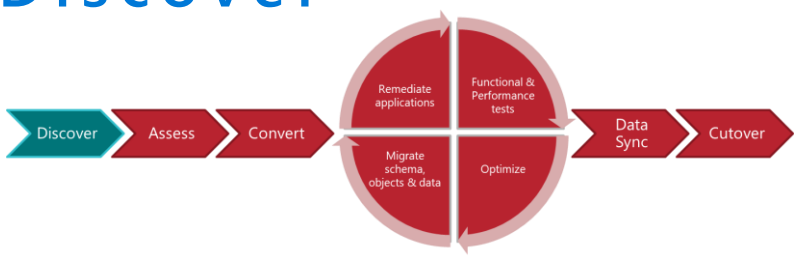
Review the Database Migration Guide for details

Link in slide: https://datamigration.microsoft.com/scenario/sql-to-sqlserver

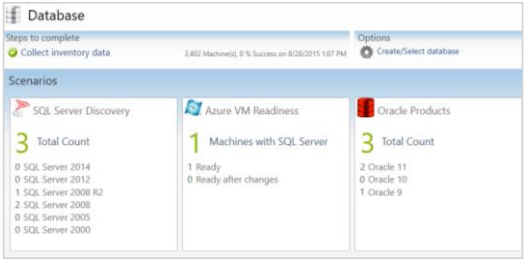
Reliable Upgrades



Discover



Discover with MAP Toolkit



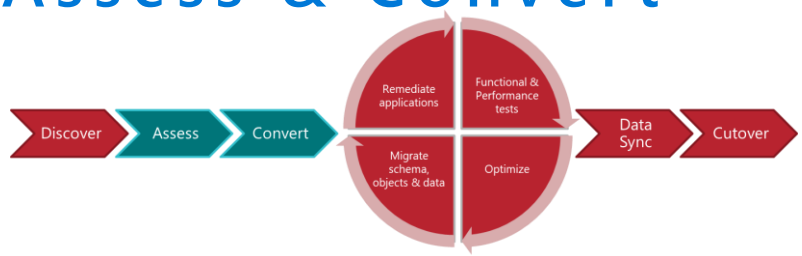
- Which SQL Server versions do I have?
- Which Editions am I running?
- Which SQL Server components are installed?
- How many cores are on each server?
- How many databases are in each instance?
- What are the sizes of all my databases?
- What are the settings for each instance and database?

The Microsoft Assessment and Planning Toolkit (MAP) is an agentless, automated, multi-product planning and assessment tool for quicker and easier desktop, server and cloud migrations. MAP can be used to provide an inventory of all your database servers, including installed components, database details, settings and capacity planning information. This Solution Accelerator provides a powerful inventory, assessment, and reporting tool to simplify the migration planning process.

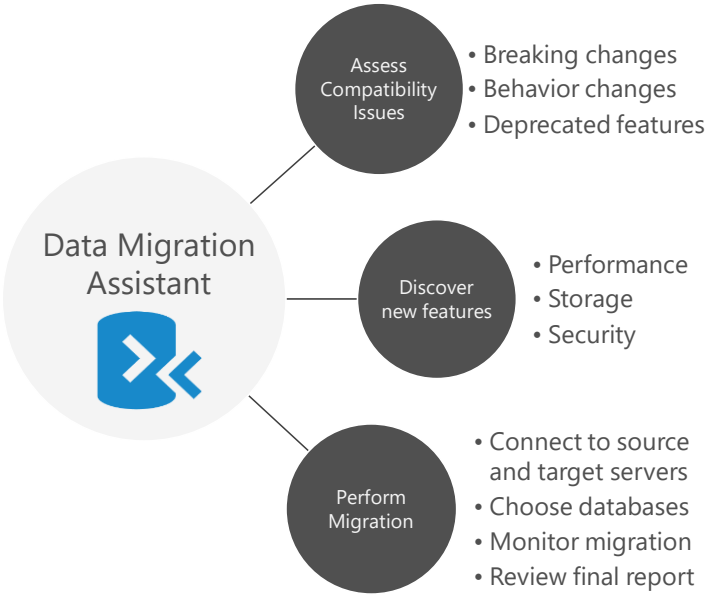
There are 8 areas that the Microsoft Assessment and Planning Toolkit (MAP) toolkit assesses:

- Cloud
- Desktop
- Server
- Desktop Virtualization
- Server Virtualization
- Database
- Usage Tracking
- Environment

Assess & Convert

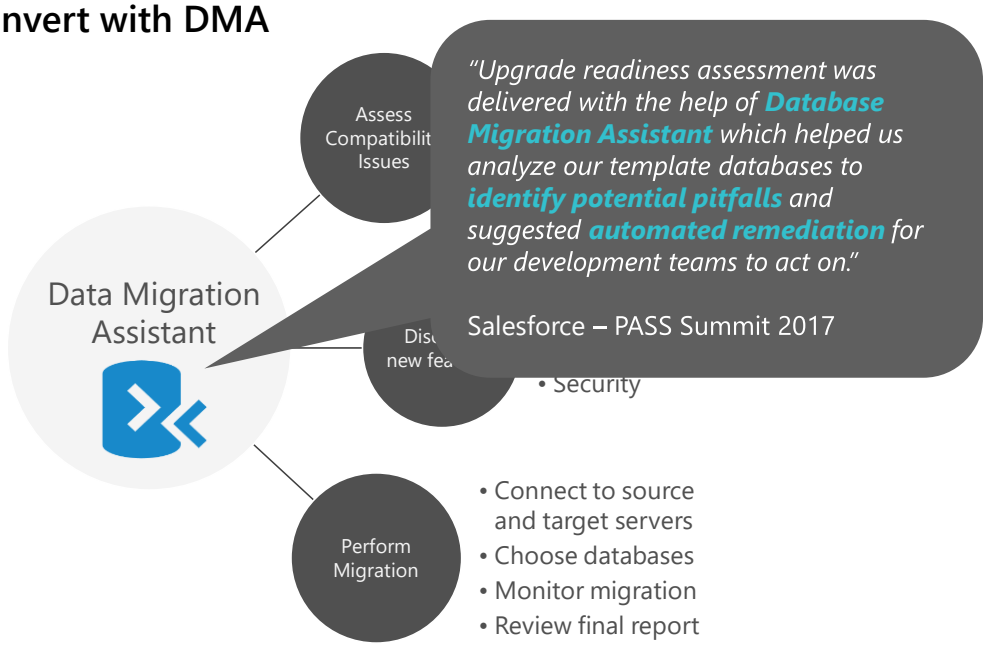


Assess & Convert with DMA



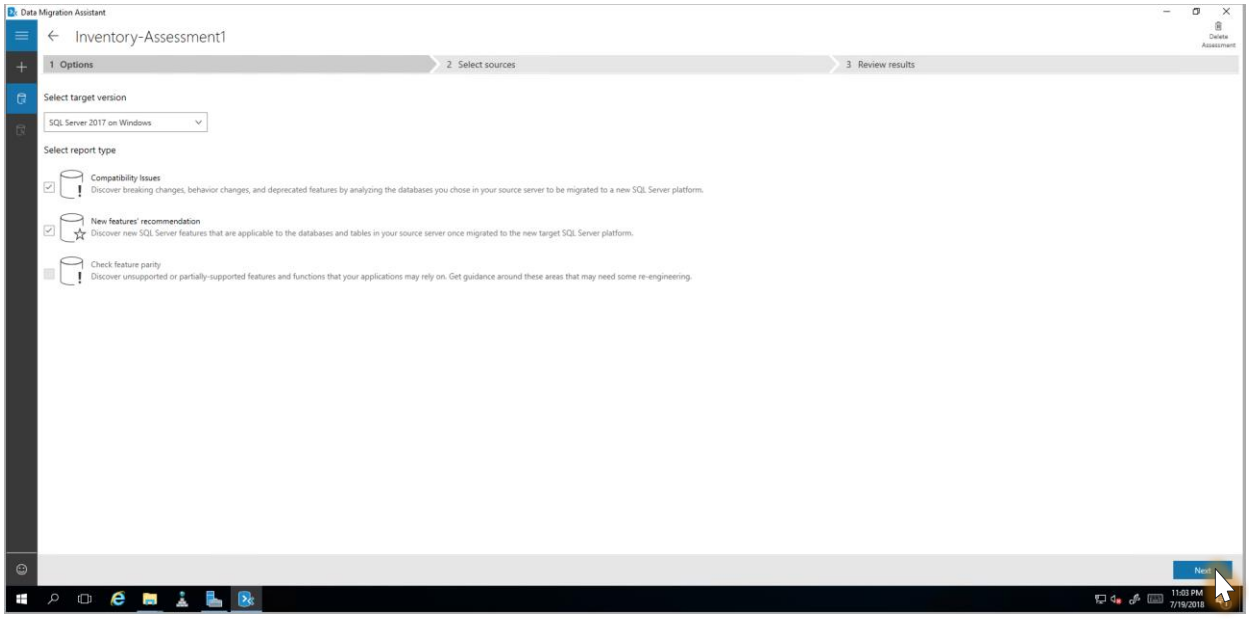
Data Migration Assistant (DMA) detects compatibility issues that can impact database functionality on your new version of SQL Server. It recommends performance and reliability improvements for your target environment. It allows you to not only move your schema and data, but also uncontained objects from your source server to your target server. Once the assessment is complete and issues have been mitigated, the DMA will automate the process of migrating your schema and data to the target instance.

Assess & Convert with DMA

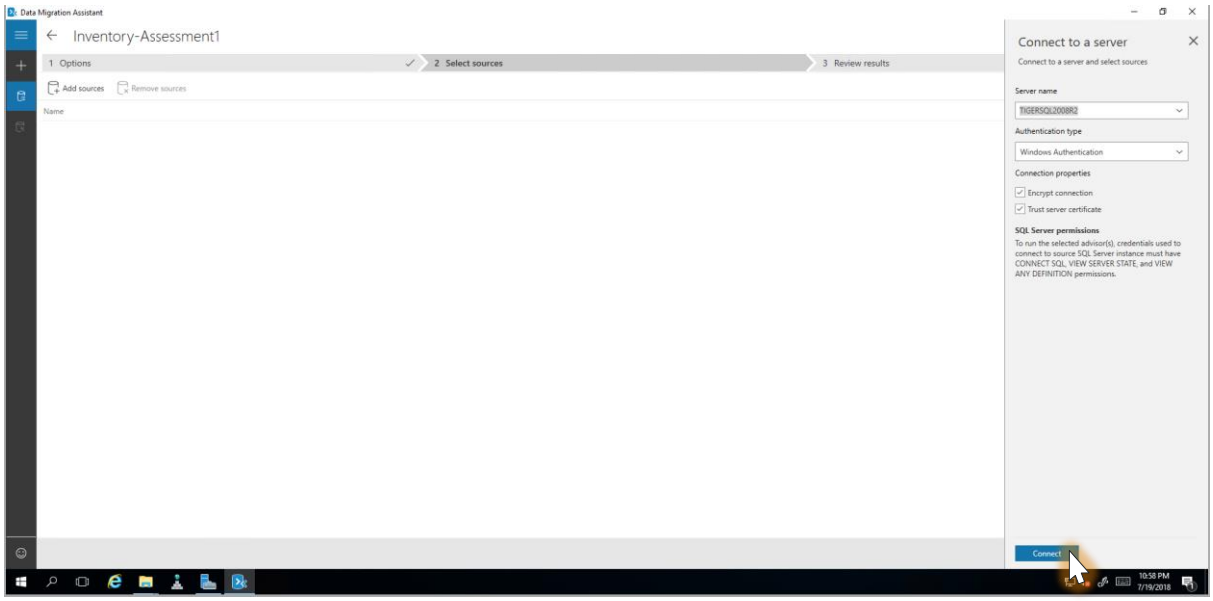


WALKTHROUGH

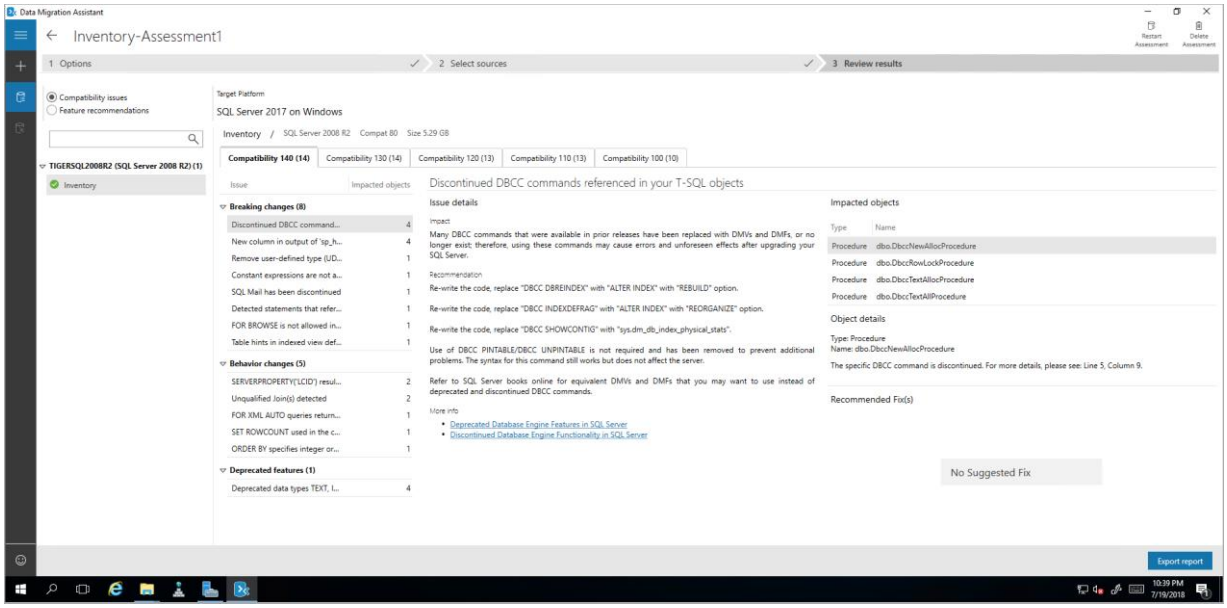
Database Migration Assistant



DMA Assessment: Options

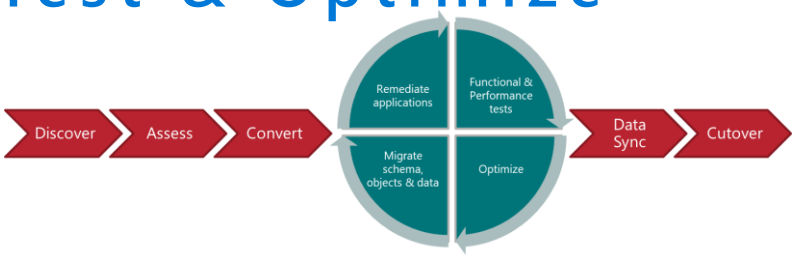


DMA Assessment: Select Sources

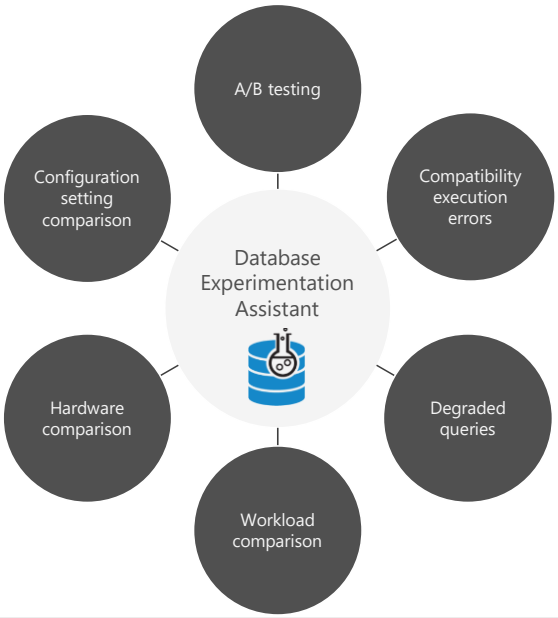


DMA Assessment: Completed

Test & Optimize



Test & Optimize with DEA



Database Experimentation Assistant (DEA) is an A/B testing solution for SQL Server upgrades. It will assist in evaluating a targeted version of SQL for a given workload. Customers who are upgrading from previous SQL Server versions (SQL Server 2005 and above) to any new version of the SQL Server will be able to use these analysis metrics.

DEA allows:

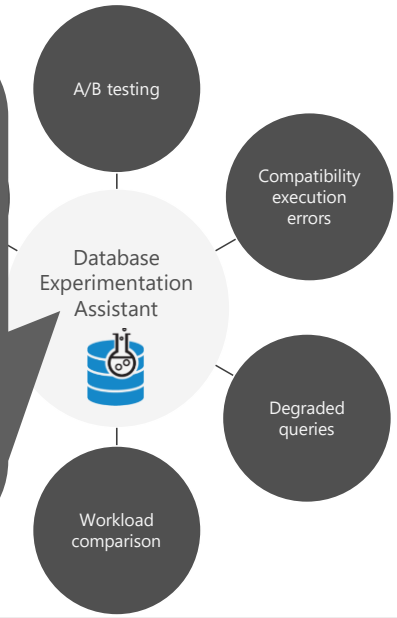
- Automated script to set up **workload capture and replay** of production database (using existing SQL server functionality Distributed Replay & SQL tracing).
- Perform **statistical analysis** on traces collected using both old and new instances.
- Visualize data through detailed **reports.**

Test & Optimize with DEA

*"**Database Experimentation Assistant** helped us perform comparison tests between our current environment and a test SQL Server 2016 environment to identify **regressions**, **breaking changes** and **performance characteristics**.*

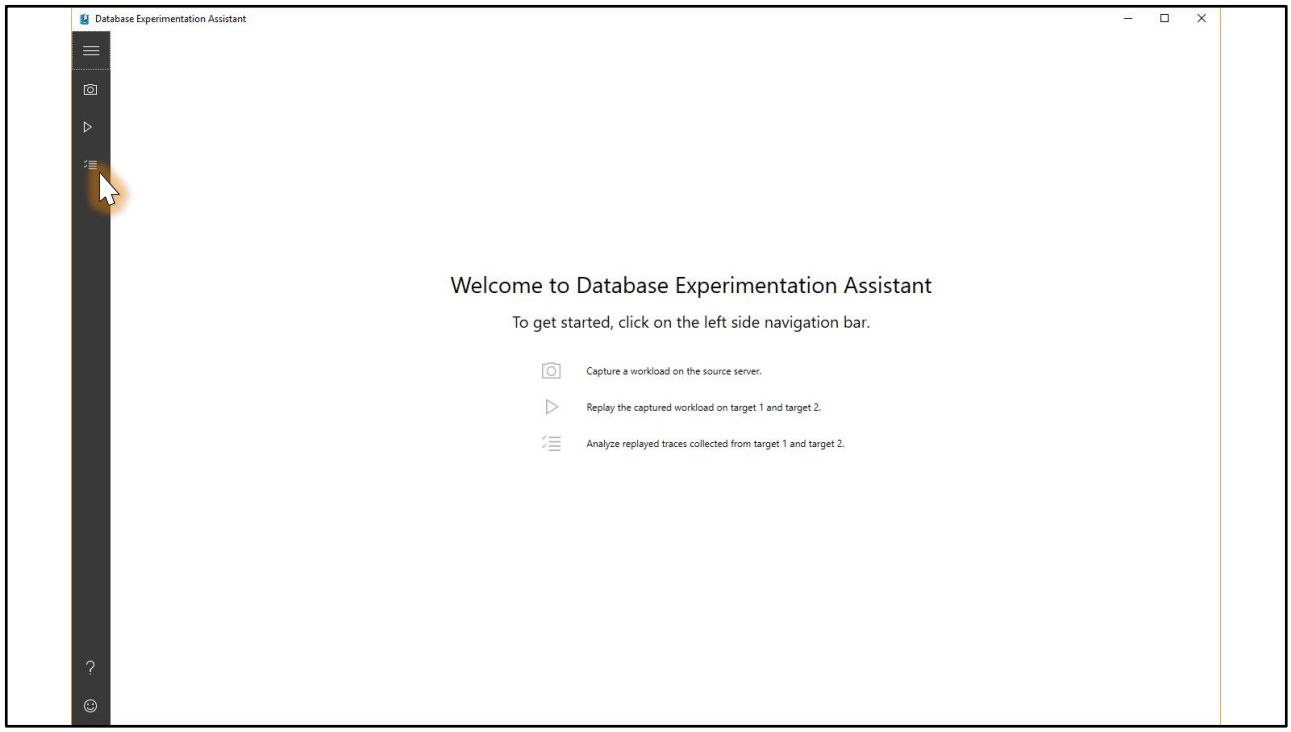
*Because of the **automation**, we were able to repeat such an exercise with multiple iterations using **different configuration settings**."*

Salesforce – PASS Summit 2017

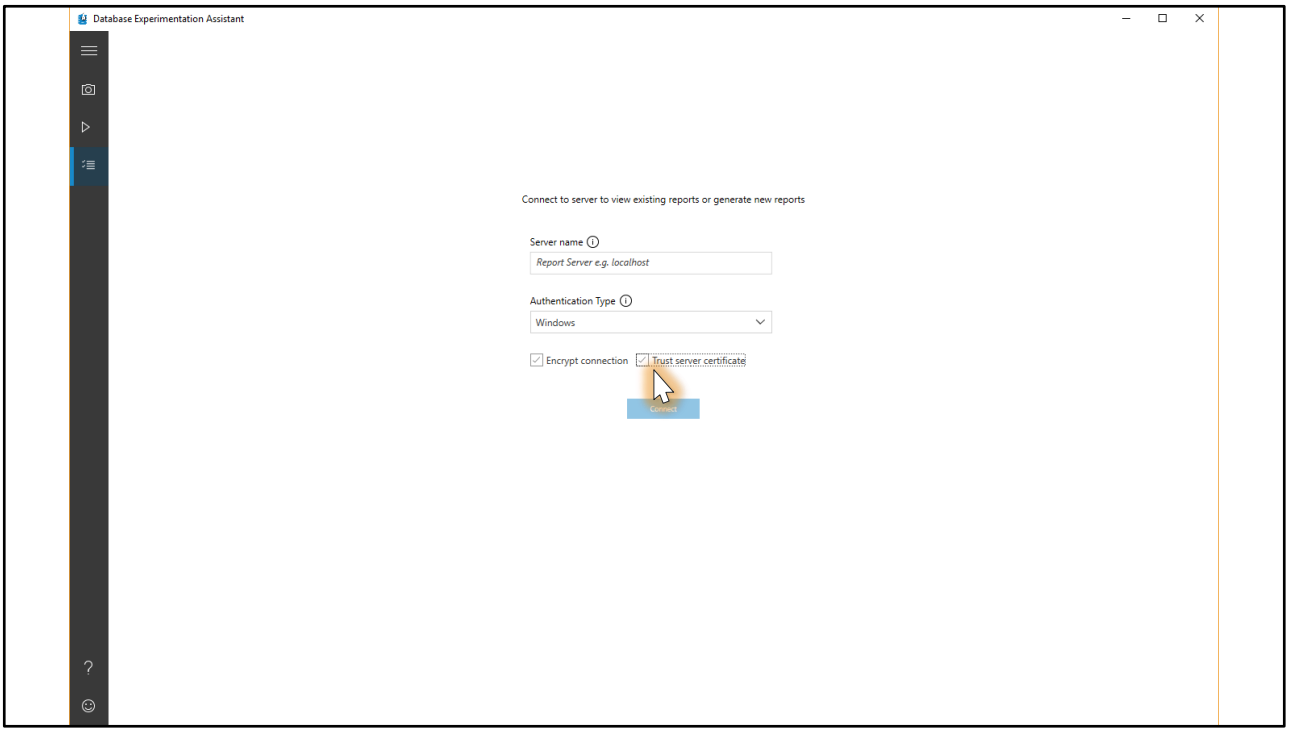


WALKTHROUGH

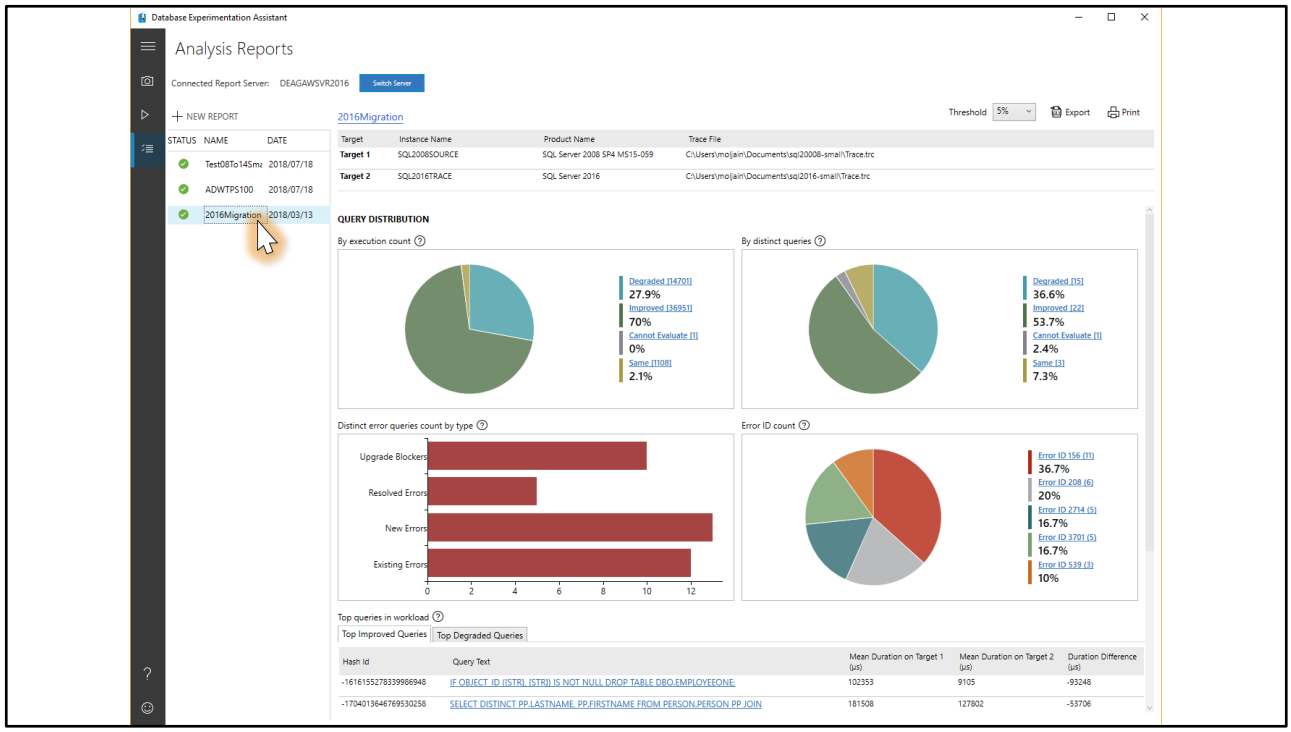
Database Experimentation Assistant



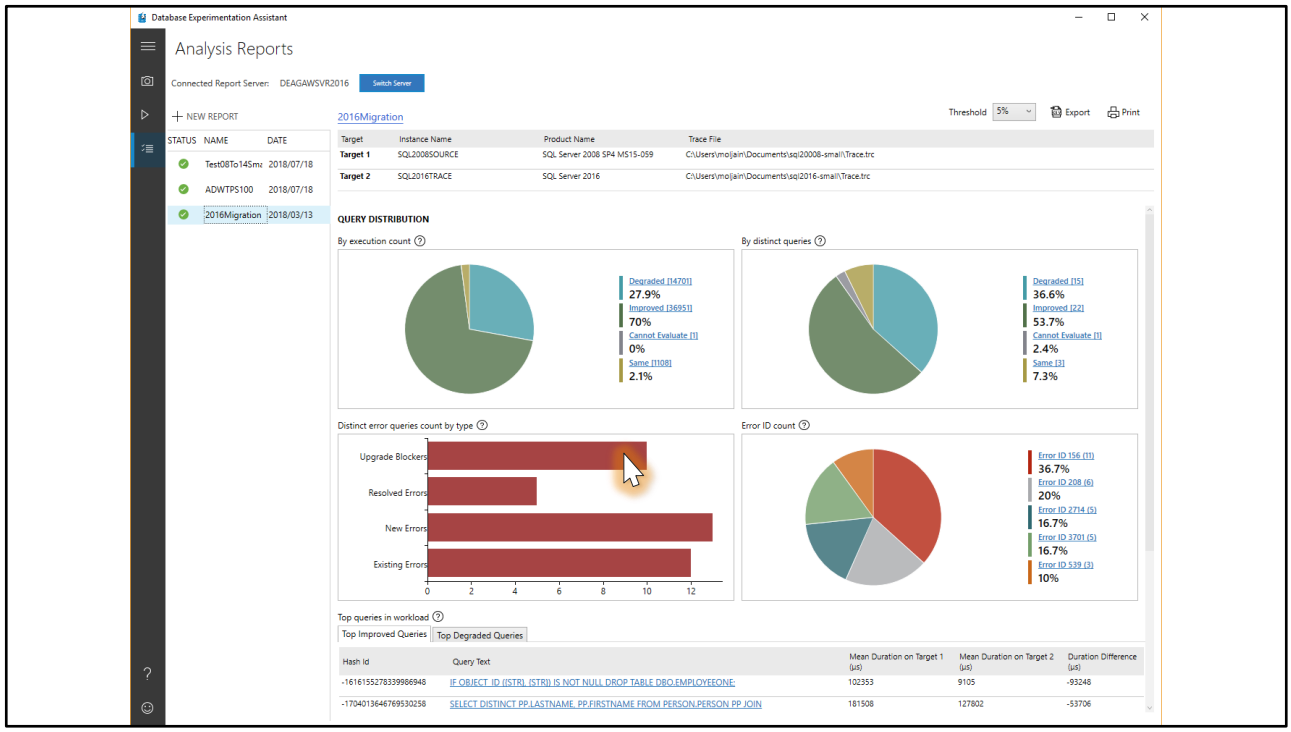
After opening, click on “Analyze traces”



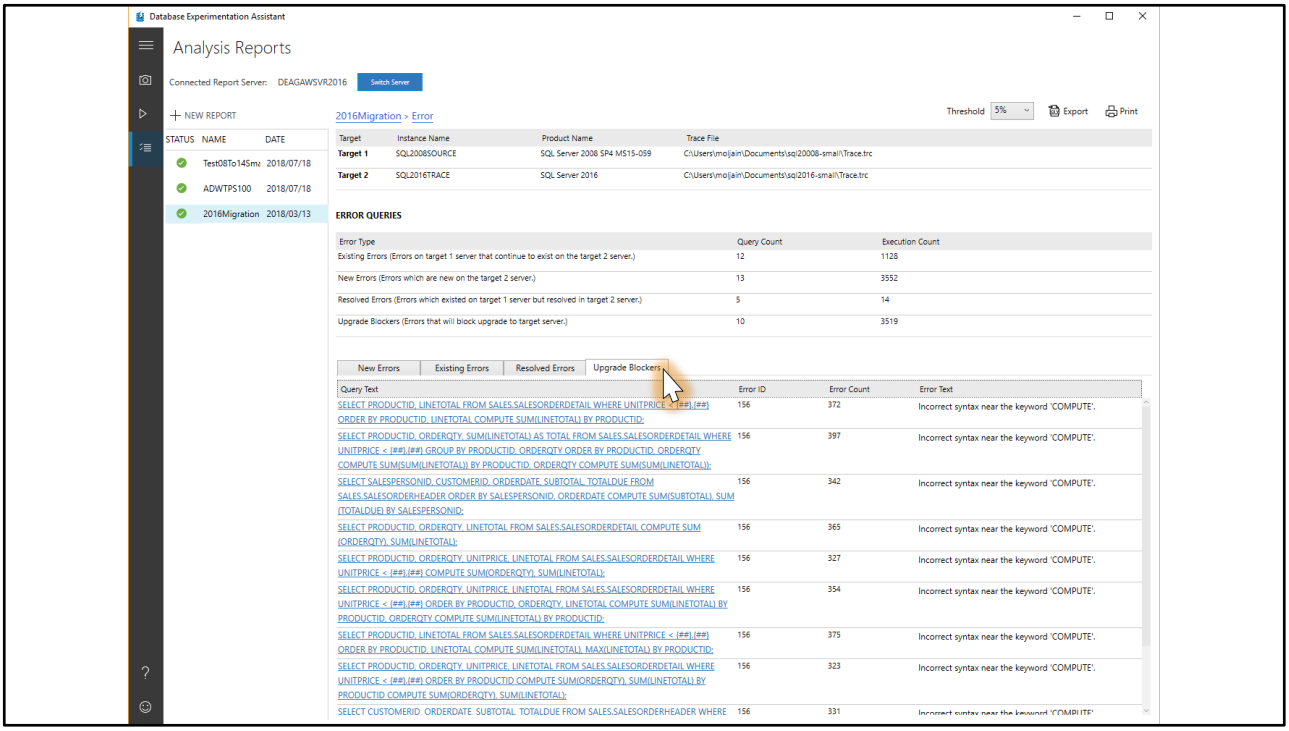
Then input server (we're using localhost) and we're using the option to trust server cert.



Once we select one of the previous replays we've done, the report shows up.

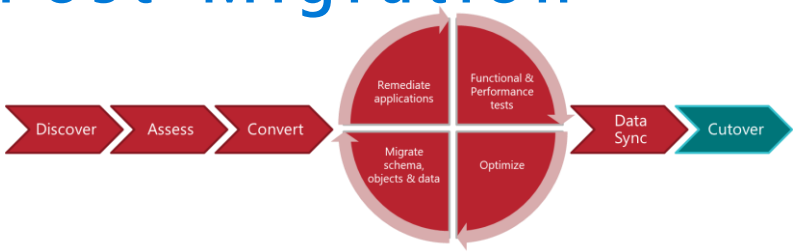


We see we have a number of potential upgrade blockers. Click on the bar to see details.



The Upgrade Blockers tab shows any syntax execution errors that existed in the replay (B) that didn't exist in the source (A). Syntax errors mapping to discontinued T-SQL are listed, in the context of the statement that produced the error. In this case, the "COMPUTE" clause was discontinued (back in SQL 2012), so this issue needs addressing.

Post-Migration



I moved the data, am I done?

SQL Server post migration step is crucial for reconciling any **data accuracy and completeness**.

But also to **uncover performance issues** with the workload.

Recommended DB Compatibility Level upgrade process:

Reference:
https://docs.microsoft.com/sql/relational-databases/performance/query-store-usage-scenarios#CEUpgrade

Why? Because of plan affecting changes – CE

CE estimates how many rows your query will likely return and is used by the Query Optimizer to generate the optimal query plan.

Most systems **benefit** from the latest CE because it is the most accurate.

Why is the latest CE more accurate?
The CE predicts how many rows your query will likely return and is used by the Query Optimizer to generate the optimal query plan.
With more accurate estimations, the Query Optimizer can usually do a better job of producing a more optimal query plan.

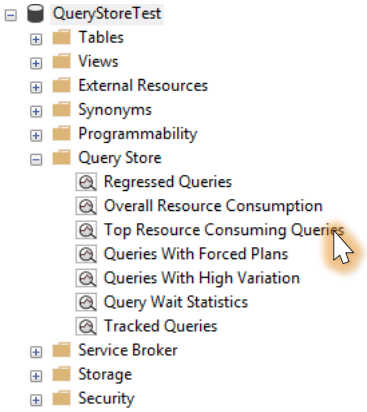
When migrating from an older versions of SQL Server to SQL Server 2014 or newer, **and** upgrading the Database Compatibility level 120 or above, a workload may be exposed to the risk of performance regression.

This is because starting with SQL Server 2014, all Query Optimizer changes are tied to the latest database compatibility level, so plans are not changed right at point of upgrade but rather when a user changes the COMPATIBILITY_LEVEL database option to the latest one. This capability, in combination with Query Store gives you a great level of control over the query performance in the upgrade process.

References:
https://docs.microsoft.com/en-us/sql/relational-databases/performance/cardinality-estimation-sql-server
https://docs.microsoft.com/en-us/sql/relational-databases/query-processing-architecture-guide

Query Store

Comprehensive query-performance information when you need it most!

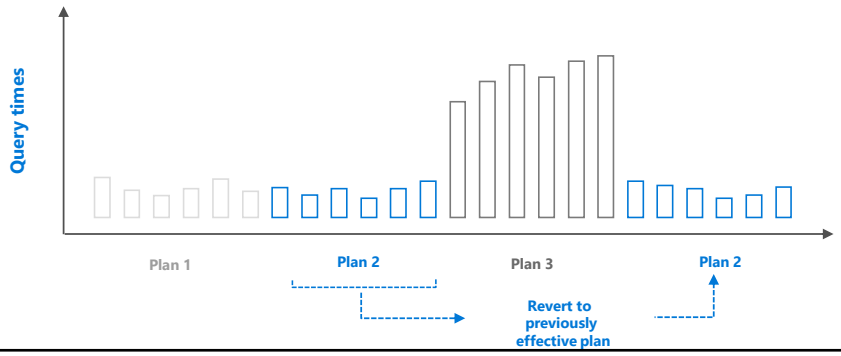


Find issues with 2 clicks.

Query Store and Automatic Plan Correction

Identifies the problematic query plan and "fixes" it to be optimal.

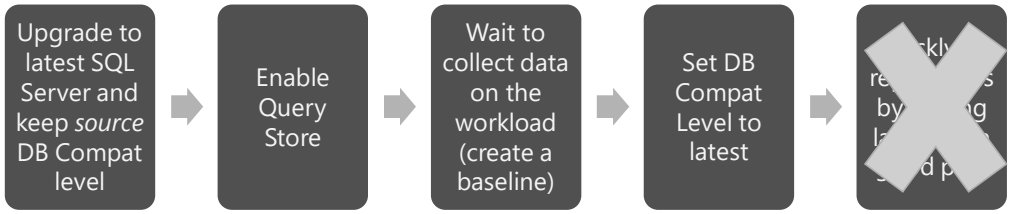
In the scope of a DB Compatibility upgrade, only works if the recommended process was followed!



DB Upgrade with Query Tuning Assistant

What if instead of choosing between current and last know good plan, we find a 3rd, better plan?

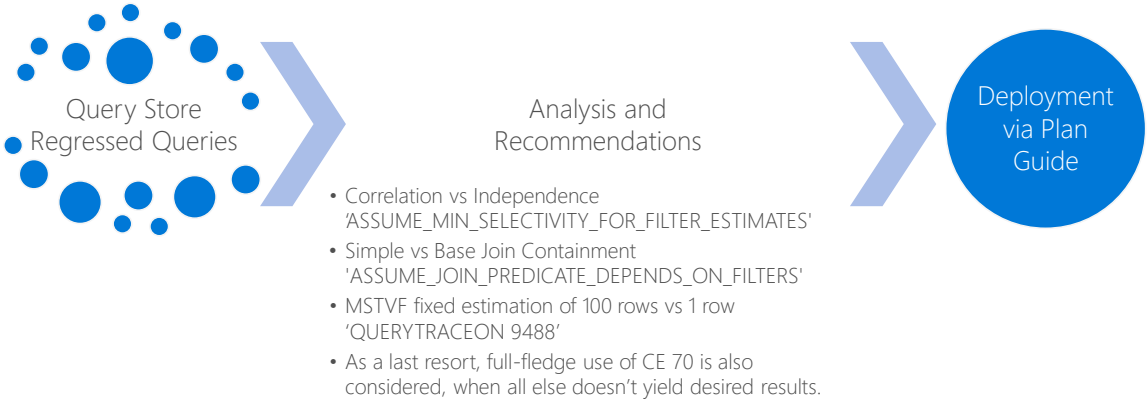
Priority 1: follow documented DB Compatibility upgrade procedure, and now you can be guided through that.



Post-migration/upgrade
Crucial to uncover query performance issues with the workload, as it runs on the newer version of SQL Server Database Engine.
User needs to follow documented DB upgrade procedure (https://docs.microsoft.com/en-us/sql/relational-databases/performance/query-store-usage-scenarios#CEUpgrade).
User database is still in before-upgrade database compatibility level, and QTA will assist in collecting baseline workload data (if none available), bump database compatibility level, collect 2nd pass of workload data, and work on any regressions found based on “Regressed Queries” QS report.

Query Tuning Assistant (QTA) Workflow

Available in SSMS v18 and Powershell (preview)



Like any hint, it forces certain behaviors that may get addressed in subsequent updates. So Microsoft recommends you only apply hints when no other option exists, and plan to revisit hinted code with every new upgrade. By forcing behaviors, you may be precluding your workload from benefiting of enhancements introduced in newer versions.

DEMO

Upgrading a database with QTA

Modernization Tools Breakdown

DMA
https://docs.microsoft.com/en-us/sql/dma/dma-assesssqlonprem
https://docs.microsoft.com/en-us/sql/dma/dma-overview
New feature recommendations include In-Memory OLTP and Columnstore, Stretch Database, Always Encrypted, Dynamic Data Masking, and Transparent Data Encryption.

DEA
Database Experimentation Assistant (DEA) is an A/B testing solution for changes in SQL Server environments (e.g. upgrade, new indexes, etc.). It assists in evaluating how the workload on your source server (current environment) will perform against your new environment. It guides you through performing an A/B test through three steps: capture, replay, and analysis.

QTA
Query Tuning Assistant (QTA) helps address some of the most common causes of CE-related performance regression, namely the following model assumption changes, starting with SQL Server 2014:

- Independence vs Correlation
- Simple Containment vs Base Containment
- TVF fixed estimation of 100 rows vs 1 row

Note: as a last resort, full-fledge use of CE 70 is also considered, when all else doesn't

yield desired results. This is done by attempting to use targeted USE HINT query hints that change these assumptions, for regressed SELECT-based queries.

Session resources

- Upgrade SQL Server
- Database Migration Guide
- Microsoft Assessment and Planning Toolkit
- Overview of Data Migration Assistant
- DEA 2.6 – Database Experimentation Assistant
- Post-migration Validation and Optimization Guide
- http://aka.ms/dbcompat (DB Compatibility Level based upgrades)

Bookmarks

SQL Server Team (Tiger) Blog	http://aka.ms/sqlserverteam
Tiger Toolbox GitHub	http://aka.ms/tigertoolbox
SQL Server Release Blog	http://aka.ms/sqlreleases
BP Check	http://aka.ms/bpcheck
SQL Server Standards Support	http://aka.ms/sqlstandards
Trace Flags	http://aka.ms/traceflags
SQL Server Support lifecycle	http://aka.ms/sqllifecycle
SQL Server Updates	http://aka.ms/sqlupdates
SQL Server Guides	http://aka.ms/sqlserverguides
SQL Feedback (New "Connect")	http://aka.ms/sqlfeedback
T-SQL Syntax Conventions	http://aka.ms/sqlconventions
SQL Server Errors	http://aka.ms/sqlerrors
SQL Performance Center	http://aka.ms/sqlperfcenter
Twitter	@mssqltiger

sqlbits

We'd love your feedback

Aka.ms/SQLBits19

Microsoft