SQLskills Immersion Event

IEAzure: Azure VMs and Azure SQL Database

Module 5: Migrating to Azure SQL Database

Tim Radney

Tim@SQLskills.com

Overview

- DTU and eDTU Calculator
- Migrating databases to Azure SQL Database
- Tuning options
- Connectivity

DTU Calculator

- **DTU and eDTU Calculator**
  - Helps determine a good starting point for DTU and eDTU sizing for your existing workload
  - Command line tool and PowerShell options
  - Collects
    - Processor - % Processor Time
    - Logical Disk – Disk Reads/sec
    - Logical Disk – Disk Writes/sec
    - Databases – Log Bytes Flushed/sec
  - http://dtucalculator.azurewebsites.net

Demo

Azure Portal: pricing and the DTU and eDTU Calculator

Migrating to Azure SQL Database

- **Where do you start?**
  - Test for compatibility
- **What next?**
  - Correct any compatibility issues
- **What then?**
  - Migrate your schema and data

Migrating to Azure SQL Database

- **Testing for compatibility**
  - SQL Server Data Tools (SSDT)
  - SqlPackage – command line tool
  - SQL Server Management Studio (SSMS)
  - Data Migration Assistant (DMA)
  - Database Migration Service (DMS)

Migrating to Azure SQL Database

- **SQL Server Data Tools (SSDT)**
  - Use most recent compatibility rules to detect SQL Database V12 incompatibilities
  - Import database schema, build a project for a SQL Database V12 deployment
  - Corrections can be synchronized back to source database
- **SqlPackage**
  - Command-line tool
  - Generates a deployment report of compatibility issues in XML
  - Always check to make sure you have the latest version!
  - https://docs.microsoft.com/en-us/sql/tools/sqlpackage?view=sql-server-2017

Migrating to Azure SQL Database

- **SQL Server Management Studio (SSMS)**
  - Export Data Tier Application in SSMS
  - Detects and reports errors to screen
  - If no errors detected, you can proceed with migration to Azure SQL Database
  - Incompatibilities can be corrected using SSMS
- **Data Migration Assistant (DMA)**
  - Replaced SQL Server 2016 Upgrade Advisor Preview
  - Standalone tool
  - Delivers scenarios to help reduce effort to upgrade
  - Can recommend performance and reliability improvements

Migrating to Azure SQL Database

- **Database Migration Service (DMS)**
  - Fully managed database migration service for both operational databases and data warehouses
  - Enables reliable and seamless migrations to the cloud at scale and minimal downtime
  - Migrate SQL Server and 3rd party databases to Azure SQL Database
  - The only tool that can migrate databases online leveraging your existing transaction log backups

Migrating to Azure SQL Database

- **Which tools can correct compatibility issues?**
  - SQL Server Data Tools (SSDT)
  - SQL Server Management Studio (SSMS)
  - Database Migration Assistance (DMA) included in DMS
- **Which tools can migrate the data?**
  - Export to BACPAC (using SSMS)
  - Transactional Replication
  - Database Migration Service (DMS)

Demo

Compatibility checking

Migrating Data to Azure SQL Database

- **SSMS Migration Wizard**
  - SQL Server 2005 and up
  - Small to medium size databases
  - Specify the location to create .bacpac
  - Handles export to .bacpac and import into Azure SQL Database automatically
  - Can use log shipping to an Azure VM to stage data
- **Database Migration Service (DMS)**
  - SQL Server 2005 and up
  - Requires a VPN or Express Route
  - Any size database within support
  - Can complete online or offline migrations

Demo

Migrating a database using SSMS Migration wizard

Migrating Data to Azure SQL Database

- **Export Data-tier Application**
  - SQL Server 2005 and up
  - Small, medium, and large size databases
  - Specify the location to create .bacpac
  - Handles export to .bacpac and import into Azure SQL Database separately
  - Can use log shipping to an Azure VM to stage data
- **Transactional replication**
  - Decreases downtime for migration
  - Requires primary keys for each published table
  - Complicated, setup distribution database, publisher and subscriber, and monitor jobs

Demo

Migrating Data – export/import

Tuning Differences

- **Instance level settings you can't change**
  - tempdb
  - Cost threshold for parallelism
  - Max degree of parallelism
  - Min and max server memory
  - Optimize for ad hoc workloads
  - Anything requiring sp_configure and RECONFIGURE
- **DBCC FREEPROCCACHE not supported**

Tuning Options

- **Increase to a higher DTU tier or vCore size**
- **This can be due to**
  - Higher I/O requirement
  - CPU demand
  - Memory demand
  - Database size
  - In-Memory OLTP limits
  - Max concurrent requests or logins
  - Max sessions

Tuning Options

- **With minor changes, your existing workload scripts should work**
  - File statistics
    - https://www.sqlskills.com/blogs/paul/how-to-examine-io-subsystem-latencies-from-within-sql-server/
    - sys.master_files does not exist, modified to use sys.databases
  - File statistics over time
    - https://www.sqlskills.com/blogs/paul/capturing-io-latencies-period-time/
    - sys.master_files does not exist, modified to use sys.databases

Tuning Options

- **With minor changes, your existing workload scripts should work**
  - Wait statistics
    - https://www.sqlskills.com/blogs/paul/wait-statistics-or-please-tell-me-where-it-hurts/
    - sys.dm_os_wait_stats contains wait statistics for the container your database is in, use sys.dm_db_wait_stats for database specific data
  - Waits over a period of time
    - https://www.sqlskills.com/blogs/paul/capturing-wait-statistics-period-time/
    - sys.dm_os_wait_stats contains wait statistics for the container your database is in, use sys.dm_db_wait_stats for database specific data
- **Getting started tuning performance in Azure SQL Database**
  - http://bit.ly/1pHB7dn
- **Glenn Berry DMVs – includes Azure SQL Database**
  - https://www.sqlskills.com/blogs/glenn/category/dmv-queries/

Query Store

- **Described as a flight data recorder**
- **Captures information about query execution**
  - Query text
  - Query plan
  - Compilation time
  - Last execution time
  - Duration, CPU, logical reads, physical reads, writes – aggregated across a time interval
- **Available in Azure SQL Database, Managed Instance, SQL Server 2016/2017/2019**
  - Call outs to specific functionality differences between SQL versions

Query Store

Enabled at the database level

Data persisted in internal tables

Cannot be enabled for master, model*, or tempdb

Requires VIEW DATABASE STATE to view data

Requires db_owner to force/un-force plans

Data is not captured on readable secondaries

Query Store – Data Captured

- Plan Store
  - Compile time and duration
  - Last execution time
  - Query text
  - Query plan
- Runtime Stats Store
  - Execution counts
  - Duration
  - CPU
  - Logical reads
  - Physical reads
  - Write
  - Memory use
  - DOP
  - Log bytes/used
  - tempdb
- Wait Stats Store
  - Wait statistics per plan

SQL Server 2017 and Azure SQL Database

Query Store – Data Not Captured

- User/login that executed the query
- Workstation from which the query was executed
- Application from which the query was executed
- Actual execution plan

Query Store Settings

- **OPERATION_MODE = [READ_WRITE | READ_ONLY]**
  - Why it matters: Do you want to capture new data?
- **QUERY_CAPTURE_MODE = [ALL | AUTO | NONE]**
  - Why it matters: Do you want to capture all queries, or just those most relevant to your workload?
- **MAX_PLANS_PER_QUERY = #**
  - Why it matters: You may have more than 200 unique plans for a query

Query Store Settings

- **MAX_STORAGE_SIZE_MB = #**
  - Why it matters: The default is 100MB, “right” value depends on multiple factors
- **CLEANUP_POLICY = (STALE_QUERY_THRESHOLD_DAY = #)**
  - Why it matters: You need to think about how much data you want to keep
- **SIZE_BASED_CLEANUP_MODE = [AUTO | OFF]**
  - Why it matters: If this is OFF and MAX_STORAGE_SIZE_MB is reached, Query Store will switch to READ_ONLY

Query Store Settings

- **DATA_FLUSH_INTERVAL_SECONDS = #**
  - Why it matters: Determines how frequently Query Store is written to disk
- **INTERVAL_LENGTH_MINUTES = #**
  - Why it matters: Affects the space needed by Query Store and the time windows across which you can analyze data
- **WAIT_STATS_CAPTURE_MODE [ON | OFF]**
  - Why it matters: Enabled by default (and enabled when upgrade to SQL 2017)
  - SQL Server 2017 and Azure SQL Database

Query Store Default Values

Setting	2016/2017	SQL DB Basic	SQL DB Standard	SQL DB Premium
QUERY_CAPTURE_MODE	ALL	AUTO	AUTO	AUTO
MAX_STORAGE_SIZE_MB	100	10	100	1024
CLEANUP_POLICY	30	7	30	30
SIZE_BASED_CLEANUP_MODE	AUTO	AUTO	AUTO	AUTO
DATA_FLUSH_INTERVAL_SECONDS	900	900	900	900

- **Tier change in Azure SQL Databases causes a change in Query Store settings, unless they were modified using ALTER DATABASE**

Query Store Settings

- **Improper configuration of these settings can cause data to be removed from Query Store before expected, or Query Store can stop collecting data entirely**
  - https://www.sqlskills.com/blogs/erin/query-store-settings/
- **Stored in sys.database_query_store_options, along with current and desired status**
- **Use Extended Events to monitor**
  - query_store_db_settings_and_state
  - query_store_db_settings_changed
  - query_store_disk_size_info
  - query_store_disk_size_over_limit

Query Store Views

Text	• sys.query_store_query_text • sys.query_store_query • sys.query_context_settings
Plan	• sys.query_store_plan
Runtime Stats	• sys.query_store_runtime_statistics • sys.query_store_runtime_statistics_interval
Wait Stats	• sys.query_store_wait_stats

Query Store System Tables

- Data resides in PRIMARY filegroup
- The Query Store data is exposed through catalog views
- The underlying tables all have primary keys, *but none have foreign keys*
  - Integrity is logically enforced within the engine
- If you find Query Store in an ERROR state (actual_state = 3), run sp_query_store_consistency_check to fix
  - Very extreme edge case
- Tables are reorganized regularly (internally managed)
  - Can use Extended Events to monitor this

Query Store – Space Needed

- **There are multiple factors to consider:**
  - How long are you keeping the data?
  - Across what interval are you capturing data?
  - What does your workload look like?
    - Number of unique queries?
    - Number of unique plans?
  - Are you capturing wait statistics (2017+)

Query Store – Space Needed

- **Default size for Azure SQL Database is either 10MB, 100MB, or 1GB (depends on tier)**
  - Counts against **total space** for the database
  - Interval for all is 60 minutes
- **My recommendation: start with 2GB and monitor**
  - current_storage_size_mb in sys.database_query_store_options
  - Extended Events
    - query_store_database_out_of_disk_space
    - query_store_disk_size_over_limit

Query Store vs. Third-Party Monitoring Tools

- **Query Store**
  - Must enable for each database
  - Retention policy affects size of data in user DB
  - Does not capture runtime parameters
  - SQL Server 2016+
  - Ability to force plans
  - Captures all queries
- **3rd Party**
  - Captures information for all databases on an instance
  - Data stored in a separate DB (on a separate instance)
  - Captures runtime parameters with queries
  - All SQL Server versions
  - Cannot force plans
  - Captures long-running queries

Query Store and Wait Statistics

- **Wait statistics are tied to the query plan and the runtime stats interval**
  - Not the same as wait stats information *in* a plan
- **Wait types are grouped into categories for simplicity**
  - Consider the resource requirements needed to track 900+ wait types
  - Slight change in approach for those who are familiar with wait stats
- **Information captured:**
  - Wait type
  - Wait time (total, average, last, min, max, stdev)

Query Store Overhead

- **Designed to have minimal overhead**
- **High volume, ad-hoc workloads may appear as though they have performance issues**
  - This can be related to how the asynchronous flush of data from memory to disk is managed internally
- **Performance optimizations added in 2017**
  - Back-ported to SQL Server 2016 SP2 CU2
    - https://support.microsoft.com/en-us/help/4340759
    - Don't let the reference to spinlocks concern you
- **Query Store Performance Overhead: What you need to know**
  - https://www.sqlskills.com/blogs/erin/query-store-performance-overhead/

Query Store Capture Mode Considerations

- Default for QUERY_CAPTURE_MODE is ALL, with AUTO recommended
  - AUTO ignores infrequent queries and any that have insignificant compile and execution duration times.
    - Thresholds (execution count, compile and runtime duration) are internally determined
- Note that Query Store still has to track what queries have executed that are insignificant
- No way to control what statements are saved in Query Store, other than with QUERY_CAPTURE_MODE
  - Some adhoc statements, if only executed once using less than *X* amount of resources, will NOT get saved in Query Store

Query Store Reports

- **Available within Management Studio**
  - Recent SSMS versions 17.X and up
- **Earlier versions of SSMS had only four (4) reports**
  - These can change with a release
- **Current reports:**
  - Regressed Queries
  - Overall Resource Consumption
  - Top Resource Consuming Queries
  - Queries With Forced Plans
  - Queries with High Variation
  - Tracked Queries
  - Query Wait Statistics - 18.x

Query Store – Using the Reports

- **Performance reports are configurable**
- **Can specify resource(s), time frame, and number of queries to return**
- **Filtering options include CPU, Duration, Execution Count, Logical Reads, Logical Writes, Memory Consumption, Physical Reads**
  - Wait Time, Log Memory, tempdb Memory, Row Count also available in SQL Server 2017
  - Avg, Min, Max, Total, StDev
- **Query Regression**
  - CPU, duration, I/O, memory

Query Store – Forcing Plans

- Query Store allows you to easily find queries with multiple plans and force one plan
- Not schema-bound
- Monitor failures with Extended events
- If a plan is no longer optimal, Query Store will continue to use it, unless you un-force it or forcing fails

A bad plan is not the one which failed, but the one which succeeded at the greatest cost.

-Anonymous DBA

Query Store – Plan Forcing

- It may not always be obvious that a plan is forced – check the actual plan and Query Store to determine
- Query performance can be different across environments for multiple reasons – including forced plans!
- Pay attention to forced plans when testing code and schema changes
  - Changing index names
  - Changing object names
- A forced plan overrides a plan guide*

Tuning Options – Automatic Tuning

- Index recommendations that have potential to improve query performance
- Analyzes SQL database's usage history
- One-click to enable
- Provides CREATE INDEX and DROP INDEX recommendations that can be applied automatically
- Can force plans automatically
- Parameterize query and fix schema recommendations
- Drop Index identifies unused and duplicate indexes (same indexed and included column, partition schema, and filters)
  - Unused = not used for a period of 93 days
- Dropped indexes will be automatically re-created if there are some queries that run slower due to the absence of the index

Tuning Options – Automatic Tuning

- Automatically created indexes are validated to make sure there is a positive gain to the workload, if not, it will revert the recommendation
  - Validation can take up to 72 hours by design
- Manually applied recommendations can take up to 48 hours to be removed from the list of recommendations
- Auto created indexes are immediately dropped when a related table or columns are dropped
- Recommendations are applied during low-usage periods < 80% DTU
- https://docs.microsoft.com/en-us/azure/sql-database/sql-database-automatic-tuning

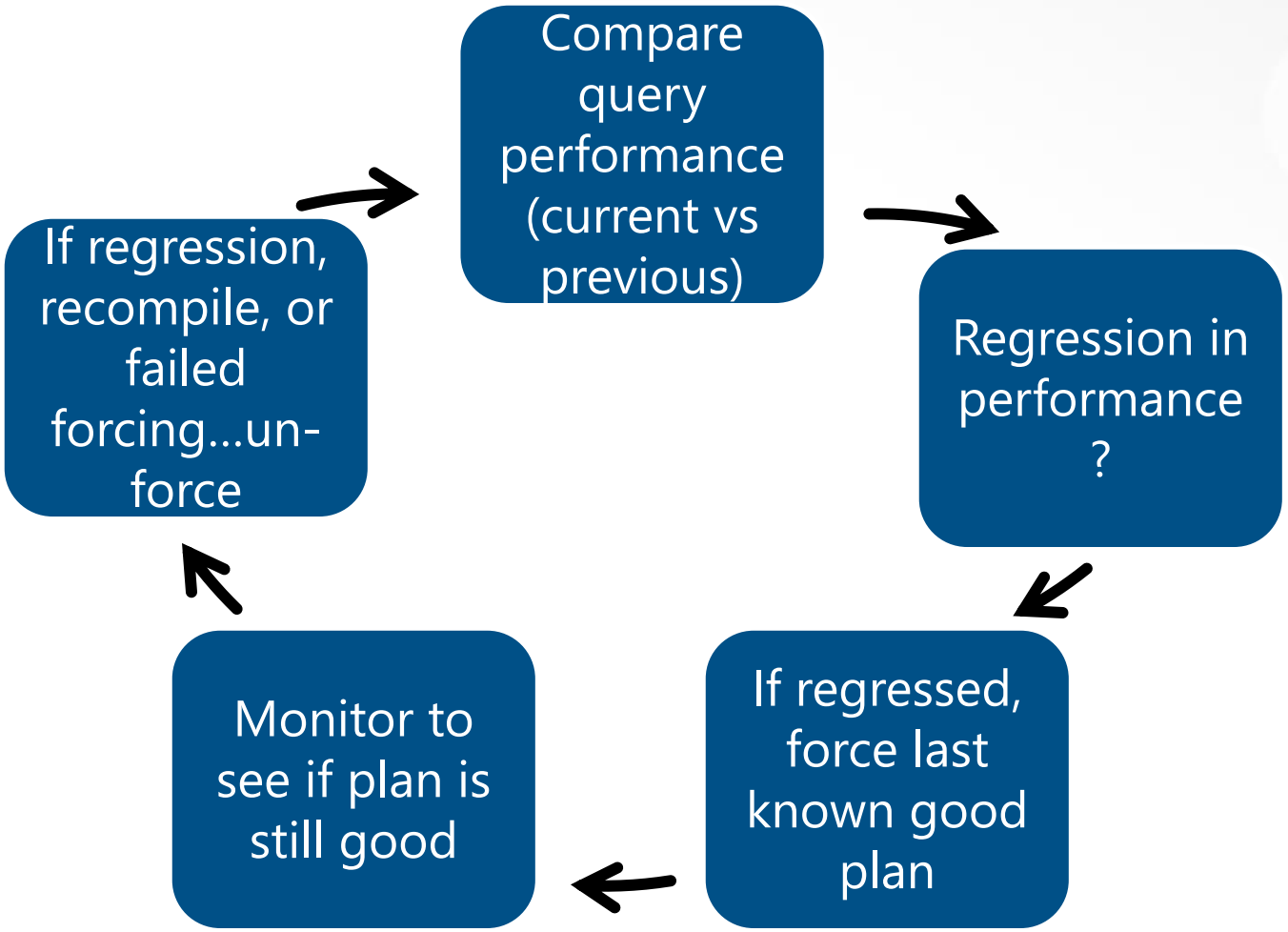
Tuning Options – Automatic Tuning

- **Notes about Indexes Created Automatically**
  - Currently, the model does merge recommendations (if multiple indexes are recommended for a table, but one index can be created to cover all options, it can create that one index)
    - However, the model is not currently intelligent enough to merge a recommended index with one that already exists
  - Indexes are not compressed
  - Within Azure SQL Database there are two times as many DROP INDEX recommendations as there are CREATE INDEX recommendations

Tuning Options – Force Plan

- **Compares current plan performance against previous plan performance**
- **Looks for significant regressions in performance**
- **If it finds a regression, will force the last good plan**
- **Continues to track whether that plan is still a good plan**
  - If not, it will un-force the plan
- **A plan will be automatically un-forced if:**
  - A recompilation occurs due to statistics or a schema change
  - Plan forcing fails
- **If you manually force a plan – either because you determined it was needed or based on a recommendation from sys.dm_db_tuning_recommendations – it will never be automatically un-forced**
- **Only plans that are forced with the Automatic Plan Correct feature will be automatically un-forced**

Tuning Options – Force Plan



Query Performance Insight

- Deeper insight into your database resource consumption
- Details on the top database queries by CPU, duration, and execution count – these are your top candidates for tuning opportunities
- Ability to drill down into the details of a query, including seeing the query text and history of resource utilization
- Annotations that display recommendations from SQL Database Advisor
- Relies on Query Store
- https://sqlperformance.com/2019/03/azure/azure-sql-database-performance-tuning-options

Demo

Auto tuning settings, Query Performance Insight

Query Store – Auto Tuning

- **Can I trust it?**
  - It is not perfect, but it has been developed with operational telemetry from Azure SQL Database implementations
  - It may not "catch" every regression you expect, and it may make a not-so-great decision
  - Its ability to recover from any "bad decision" is highly reliable as there is continual validation of forced plans and automatic back-off logic built-in

Connectivity

- Server name – abcxyx.database.windows.net
- Connection strings available
  - ADO.NET
  - JDBC
  - ODBC
  - PHP
- Firewall rules
- Login and password

Demo

Configuring connection strings, firewall rules, and creating users

Key Takeaways

- The DTU Calculator is a great start to measure how much an instance would require in DTU or eDTUs prior to a migration
- Several tools exist to help detect which databases are compatible with Azure SQL Database
- Some of those tools also allow you to remediate the issues
- Multiple ways exist to migrate actual data, unfortunately, backup and restore is not one of them
- Tuning is a lot like you do for local SQL Server instances, with some minor differences
- Before you can connect, you will need firewall access, login and password, and the server's name

Review

- DTU Calculator
- Migrating databases to Azure SQL Database
- Tuning options
- Connectivity

References

- DTU Calculator
  - http://dtucalculator.azurewebsites.net
- SQL Server Data Tools
  - http://bit.ly/2tfN4c5
- SqlPackage
  - http://bit.ly/2wVLYZg
- Data Migration Assistant
  - http://bit.ly/2c5QOHh

Questions?