Microsoft | intel

Set Yourself Up for Performance Success on Azure SQL Managed Instance

Nevena Nikolic, Product Manager

Speaker

Nevena Nikolic

Product Manager
Microsoft

Nevena Nikolic is a Cambridge University alumnus; a mechanical engineer by BA, an energy engineer by MEng and a product manager in SQL Managed Instance team by fate.

Nevena is primarily focused on performance improvements and new features development to help users tune and monitor the performance of their instances.

Prior to joining Microsoft, she worked as a quantitative developer, developing and implementing mathematical models for finance software.

in linkedin.com/in/nevena-nik/

✉ nnikolic@microsoft.com

Agenda

- **Part I**: Select the right combination of **compute & storage** to maintain/improve performance
- **Part II**: Improve performance with **new TempDB configurations & In-Memory technologies**
- **Part III**: **Monitor and tune performance** for your managed instance
- **Part IV**: Q&A ☺

Part I:

Select the right combination of **compute & storage** to maintain/improve performance

First of all: Perf Baseline

Performance baseline is a set of parameters that define your workload on your on-premises servers.

How-to:

- Define your business requirements for uptime and data redundancy
- Measure and document
  - min/average/max duration and CPU usage for the queries
  - performance metrics (average/max CPU usage, average/max disk IO latency, throughput, IOPS, average / max page life expectancy, and average max size of tempdb).

Perf Baseline – how to (I):

Identify and document

Average and peak values on your source system:

- CPU usage
- memory usage
- IO usage

For the dominant and the most critical queries in your workload:

- average and max duration
- CPU usage
- Execution plans

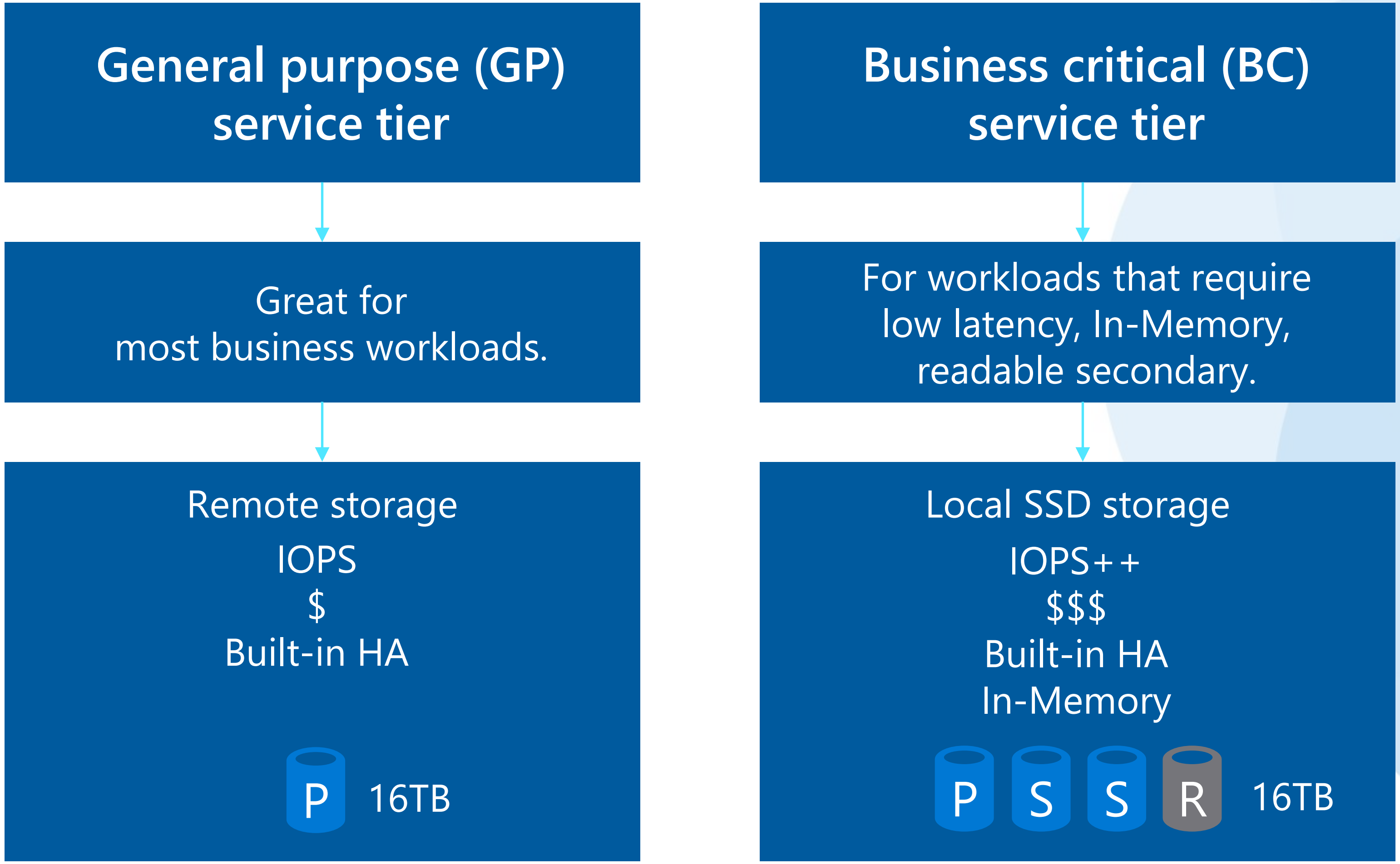
Visit: [Establish a Performance Baseline - SQL Server | Microsoft Learn]

Tips&tricks: Perf Baseline – how to (II):

- Monitor CPU usage
  - Use Server/Performance dashboard reports in SQL Server Management studio
    - Reports > Performance Dashboard
    - Reports > Standard Reports > Performance - Top Queries by Average CPU time
  - Query Store (SQL Server 2016+)
    - Database > Query Store > Top Resource Consuming Queries

- Monitor memory usage
  - by different components such as buffer pool, plan cache, column-store pool, In-Memory OLTP, etc.
  - Find average and peak values of the Page Life Expectancy memory performance counter
  - Note: Do not monitor usage of available memory [%] – this is a common mistake. Better metric for memory usage monitoring is ‘Page Life Expectancy’ (PLE) performance counter
  - If the results show that the PLE is permanently smaller on Managed Instance, you should upscale your instance.
- Monitor disk IO usage
  - sys.dm_io_virtual_file_stats view or performance counters.
  - Measure IO latency of the file subsystem to choose between service tiers.

Choose SQL Managed Instance service tier



General purpose (GP) service tier

Great for most business workloads.

Remote storage
IOPS
$
Built-in HA

P 16TB

Business critical (BC) service tier

For workloads that require low latency, In-Memory, readable secondary.

Local SSD storage
IOPS++
$$$
Built-in HA
In-Memory

P S S R 16TB

Resource limits

- Memory
- Max Log Size
- I/O throughput and latency
- Size of TempDB
- Max concurrent workers
- Backup Retention

Visit: Resource limits - Azure SQL Managed Instance | Microsoft Learn

File IO characteristics in General Purpose tier

File size	**>=0 and <=129 GiB**	**>129 and <=513 GiB**	**>513 and <=1025 GiB**	**>1025 and <=2049 GiB**	**>2049 and <=4097 GiB**	**>4097 GiB and <=8 TiB**
IOPS per file	500	2300	5000	7500	7500	12,500
Throughput per file	100 MiB/s	150 MiB/s	200 MiB/s	250 MiB/s	250 MiB/s	250 MiB/s

Note: The instance-level limit on the max log write throughput

Log write throughput limit (per instance)	4.5 MiB/s per vCore Max 120 MiB/s per instance 22 - 65 MiB/s per DB (depending on log file size)

Note : If you see WRITELOG or PAGEIOLATCH wait statistics in the wait statistics analysis, **increasing the file size should help**.

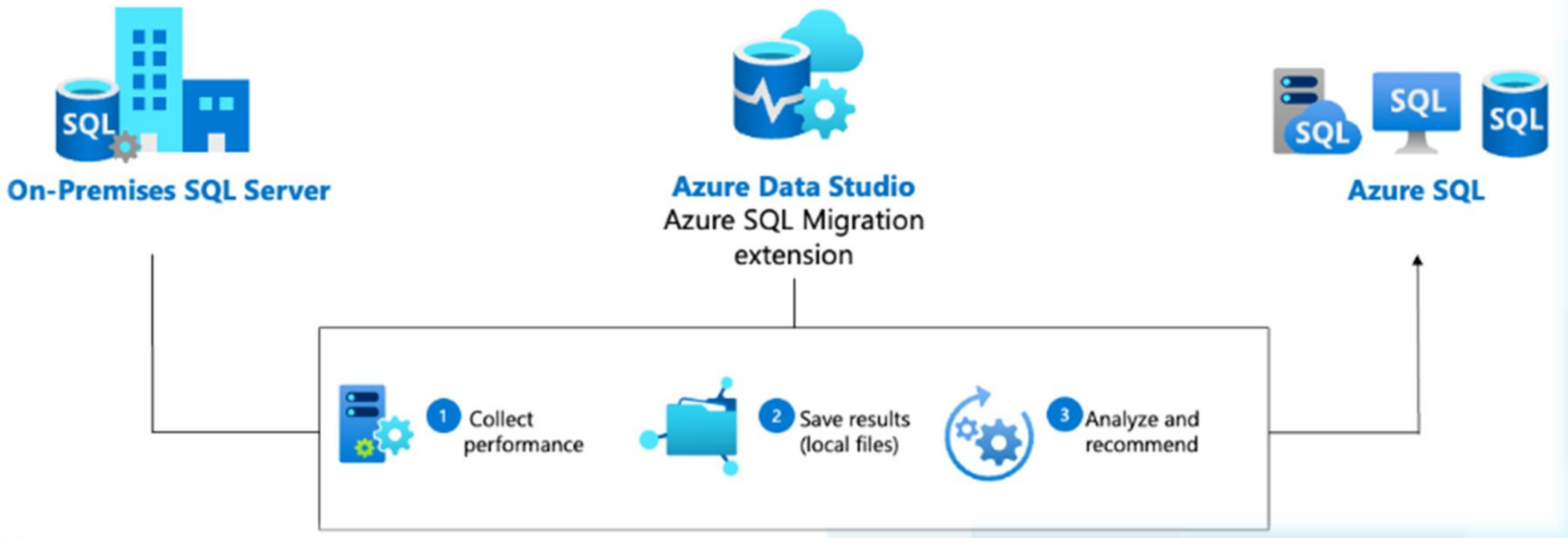
Visit: Increase data file size to improve HammerDB workload performance on General Purpose Managed Instance - Microsoft Community Hub

Choose SQL MI hardware configuration

Backed by Intel® Xeon® Scalable processors

Standard-series (Gen 5)	Premium-series	Premium-series Memory-Optimized
CPU: Intel® Xeon® 1st Generation and Intel® Xeon® 2nd Generation Scalable processors, 2.3-2.5 GHz	CPU: Intel® Xeon® 3rd Generation Scalable processors, 2.8 GHz	CPU: Intel® Xeon® 3rd Generation Scalable processors, 2.8 GHz
vCore range: 4 – 80	vCore range: 4 – 80	vCore range: 4 - 64
Memory / vCore: 5.1 GB	Memory / vCore: 7 GB	Memory / vCore: 13.6 GB
Max instance memory: 408GB	Max instance memory: 560GB	Max instance memory: 870 GB
Max instance storage	Max instance storage	Max instance storage
General Purpose: 16 TB	General Purpose: 16 TB	General Purpose: 16 TB
Business Critical: 4 TB	Business Critical: 5.5 TB	Business Critical: 16 TB

Azure SQL SKU recommendations



Prerequisites:

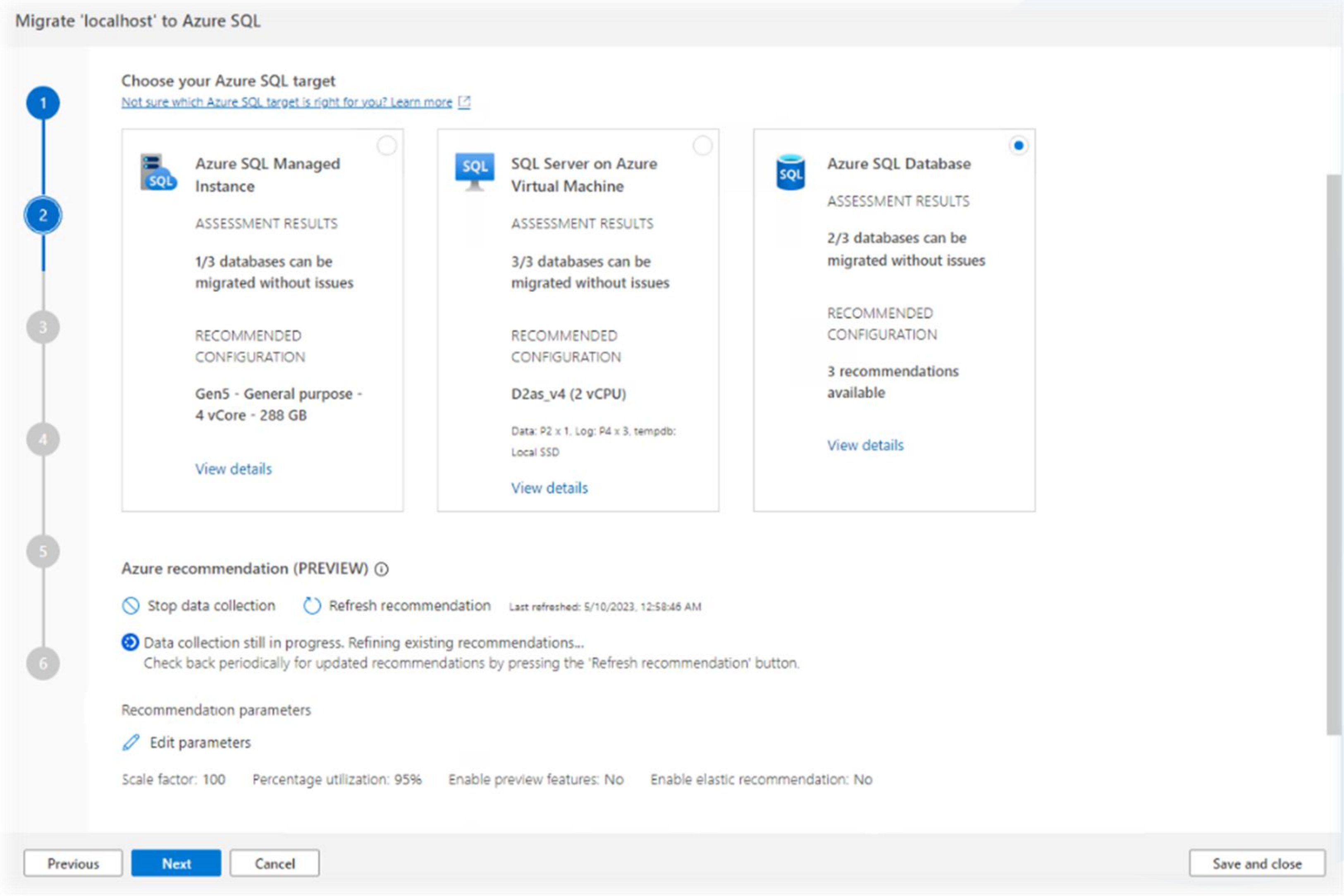
•Download and install Azure Data Studio.

•Install the Azure SQL Migration extension from Azure Data Studio Marketplace.

•Ensure that the login you use to connect the source SQL Server instance, has the minimum permissions.

Get Azure SQL SKU recommendations (Data Migration Assistant) - SQL Server | Microsoft Learn

Azure SQL SKU recommendations



Choose your Azure SQL target

Not sure which Azure SQL target is right for you? Learn more

○ **Azure SQL Managed Instance**

ASSESSMENT RESULTS

1/3 databases can be migrated without issues

RECOMMENDED CONFIGURATION

Gen5 - General purpose - 4 vCore - 288 GB

View details

○ **SQL Server on Azure Virtual Machine**

ASSESSMENT RESULTS

3/3 databases can be migrated without issues

RECOMMENDED CONFIGURATION

D2as_v4 (2 vCPU)

Data: P2 x 1, Log: P4 x 3, tempdb: Local SSD

View details

● **Azure SQL Database**

ASSESSMENT RESULTS

2/3 databases can be migrated without issues

RECOMMENDED CONFIGURATION

3 recommendations available

View details

Azure recommendation (PREVIEW)

Stop data collection   Refresh recommendation   Last refreshed: 5/10/2023, 12:58:46 AM

Data collection still in progress. Refining existing recommendations...
Check back periodically for updated recommendations by pressing the 'Refresh recommendation' button.

Recommendation parameters

Edit parameters

Scale factor: 100   Percentage utilization: 95%   Enable preview features: No   Enable elastic recommendation: No

Visit: [Get Azure recommendations for your SQL Server migration | Microsoft Learn]

Tips&tricks: Compare environment settings on source and target instance

Compare:

- Technical characteristics (cores, memory, IO)
- Server/database properties (compatibility levels, maxdop, cardinality estimator, encryption, etc.)
- Trace flag settings
- Tempdb settings (number of files, encryption)

How-to:

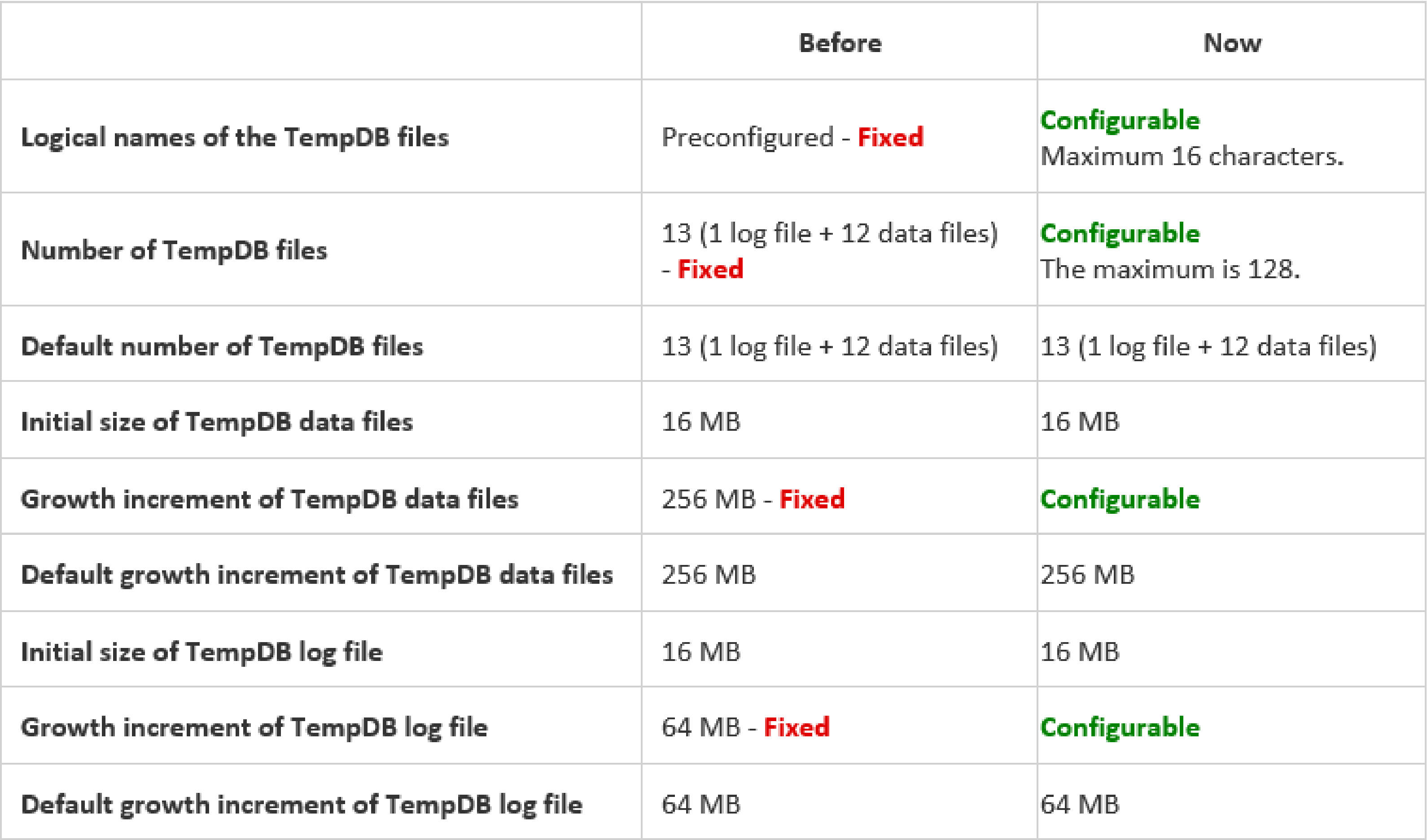
- Get the instance properties Get-properties.sql
- Compare the environment settings Compare-properties.sql

Note: You don't need to take some immediate action whenever you see some difference or assume that every difference listed here can the cause the performance issue.

Part II:

Improve performance with **new TempDB configurations & In-Memory technologies**

TempDB Configurations



	Before	Now
Logical names of the TempDB files	Preconfigured - **Fixed**	**Configurable** Maximum 16 characters.
Number of TempDB files	13 (1 log file + 12 data files) - **Fixed**	**Configurable** The maximum is 128.
Default number of TempDB files	13 (1 log file + 12 data files)	13 (1 log file + 12 data files)
Initial size of TempDB data files	16 MB	16 MB
Growth increment of TempDB data files	256 MB - **Fixed**	**Configurable**
Default growth increment of TempDB data files	256 MB	256 MB
Initial size of TempDB log file	16 MB	16 MB
Growth increment of TempDB log file	64 MB - **Fixed**	**Configurable**
Default growth increment of TempDB log file	64 MB	64 MB

Note:

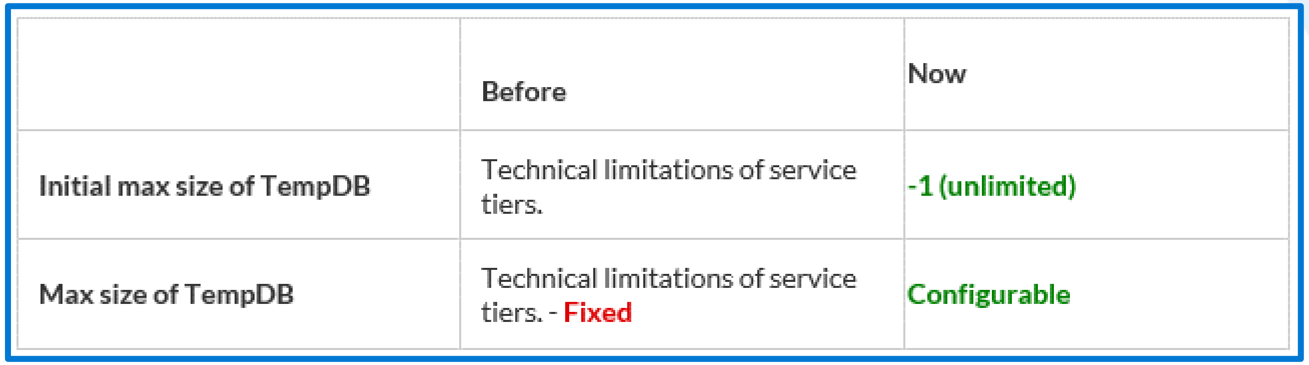
- The maximum number of TempDB files is **128**.
- Logical file name maximum **16** characters
- You **do not have to restart** the server after adding/removing new files
- We strongly suggest setting the **growth increments the same across all TempDB data files**.

Visit: Improve your SQL Managed Instance performance with new TempDB configurations - Microsoft Community Hub

TempDB Configurations – there is more!

- **TempDB max size** - the limit after which TempDB cannot further grow.
- Technical limitations
- Manually imposed limitations [NEW]

Service tier	General Purpose	Business Critical
Max tempdb database size	Limited to 24 GB/vCore (96 - 1,920 GB) Log file size is limited to 120 GB.	Up to currently available instance storage size. Log file size is limited to 2 TB.



	Before	Now
Initial max size of TempDB	Technical limitations of service tiers.	-1 (unlimited)
Max size of TempDB	Technical limitations of service tiers. - Fixed	Configurable

Visit: Configure your TempDB max size in Azure SQL Managed Instance - Microsoft Community Hub

In-Memory technologies

- Stored and processed in memory.
- Leverage the speed and efficiency of memory access.
- Designed to complement traditional disk-based storage and provide additional performance benefits for specific workloads.

Note: Available in Business Critical tier.

In-mem techs in SQL Managed Instance are:

- In-Memory OLTP (Hekaton)
- Memory-Optimized Filegroups
- Natively Compiled Stored Procedures
- Columnstore Indexes

Visit: In-memory technologies - Azure SQL | Microsoft Learn

In-Memory OLTP (Hekaton)

Allows you to create **memory-optimized tables** that reside in memory and are optimized for high-performance transaction processing.

In-Memory OLTP tables are:

- ✓ fully durable
- ✓ lock-free
- ✓ latch-free
- ✓ with optimistic concurrency control mechanisms

➡ Convert performance-critical tables to In-Memory OLTP tables

Note: Resolve Schema Limitations.
Note: Best suited for tables that undergo high rates of data modifications and require frequent access

WITH (MEMORY_OPTIMIZED = ON)

Visit: [Overview and Usage Scenarios - SQL Server | Microsoft Learn]

Memory-Optimized Filegroups

Specifically designed filegroups to store and manage the data and indexes of In-Memory OLTP tables

Reside in memory (eliminating disk I/O and reducing latency)

Can be configured to provide the necessary memory capacity and durability for your in-memory data

➡ Before converting tables, create a memory-optimized filegroup

Natively Compiled Stored Procedures

Allows you to write **stored procedures that are compiled to native code** and executed directly in memory

➡ Rewrite critical stored procedures as natively compiled stored procedures.

Note: You must review the code of the critical stored procedures and modify it to comply with the requirements of natively compiled stored procedures. Consider:

- Replace unsupported T-SQL syntax
- Eliminate certain operation
- Address dependencies on disk-based tables

NATIVE_COMPILATION

Columnstore Indexes

Store and process data column-wise. Well-suited for data warehousing and reporting scenarios

➡ Consider implementing Columnstore indexes for analytical workloads or reporting queries or hybrid (TA) processing

Can reduce your storage footprint up to 10 times!

```
CREATE CLUSTERED COLUMNSTORE INDEX index_name ON table_name;
```

- replaces the entire table storage and is suitable for tables primarily used for reporting and analytical queries (OLAP)

```
CREATE NONCLUSTERED COLUMNSTORE INDEX index_name ON table_name;;
```

- created alongside an existing table structure and is suitable for tables that are used for both analytical and transactional workloads (HTAP)

Visit: [Columnstore indexes: Overview - SQL Server | Microsoft Learn]

Optimize Columnstore Index

Delta Store Compression:
periodically run the ALTER INDEX REORGANIZE statement to compress the Delta Store into the columnstore segment.

Data Compression:
compression level can be set to NONE, ROW, PAGE, or COLUMN. Experiment!

Part III:
Monitor and Tune performance for your Azure SQL Managed Instance

Monitoring and tuning

Monitoring solutions:

- Monitor using DMVs
- Monitor using query store
- SQL Insights (preview) in Azure Monitor
- Azure SQL Analytics (preview) using Azure Monitor Logs

Visit: Monitoring and performance tuning - Azure SQL Database | Microsoft Learn

Monitor & Tune with DMVs *(Dynamic Management Views)*

- Detectable types of query performance bottlenecks in Azure SQL MI

Running-related problems:

- **compilation problems** resulting in a suboptimal query plan or
- **execution problems** related to insufficient or overused resources.

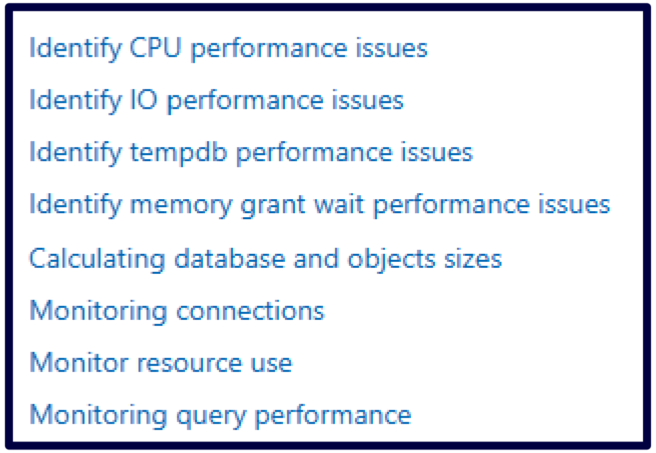
Waiting-related problems:

- Locks (blocking)
- I/O
- Tempdb contention
- Memory grant waits

Note: use SQL Server DMVs to detect these types of performance bottlenecks.

Monitor with DMVs

- Requires **VIEW SERVER STATE** permissions → GRANT VIEW SERVER STATE TO database_user;



Identify CPU performance issues
Identify IO performance issues
Identify tempdb performance issues
Identify memory grant wait performance issues
Calculating database and objects sizes
Monitoring connections
Monitor resource use
Monitoring query performance

- sys.dm_db_tuning_recommendations view

Visit: Monitor performance using DMVs - Azure SQL Managed Instance | Microsoft Learn

Tips&tricks: TempDB performance issues

Identify tempdb performance issue:

PAGELATCH_* + use sys.dm_exec_requests to confirm that the wait_resource value begins with 2:x:y where 2 is the database ID (tempdb), x is the file ID, and y is the page ID

Identify top queries that use table variables and temporary tables:

```
SELECT plan_handle, execution_count, query_plan
INTO #tmpPlan
FROM sys.dm_exec_query_stats
           CROSS APPLY sys.dm_exec_query_plan(plan_handle);
GO

WITH XMLNAMESPACES('http://schemas.microsoft.com/sqlserver/2004/07/showplan' AS sp)
SELECT plan_handle, stmt.stmt_details.value('@Database', 'varchar(max)') 'Database', stmt.stmt_details.value('@Schema',
'varchar(max)') 'Schema', stmt.stmt_details.value('@Table', 'varchar(max)') 'table’
INTO #tmp2
FROM(SELECT CAST(query_plan AS XML) sqlplan, plan_handle FROM #tmpPlan) AS p
           CROSS APPLY sqlplan.nodes('//sp:Object') AS stmt(stmt_details);
GO

SELECT t.plan_handle, [Database], [Schema], [table], execution_count
FROM(SELECT DISTINCT plan_handle, [Database], [Schema], [table]
FROM #tmp2
WHERE [table] LIKE '%@%' OR [table] LIKE '%#%') AS t
           JOIN #tmpPlan AS t2 ON t.plan_handle=t2.plan_handle;
```

Tips&tricks: Query Performance Insight library

Example I:

Use DMVs to monitor CPU:

- ➢ sys.dm_os_ring_buffers or
- ➢ master.sys.server_resource_stats

Use Query Performance Insight library for easier analyzing:

- ➢ select * from qpi.cpu_usage

Example II:

Use DMVs to monitor wait stats:

- ➢ **Global/Instance-level wait statistics** using sys.dm_os_wait_stats
- ➢ **Session-level wait statistics** using sys.dm_exec_session_wait_stats

Use QPI library for easier analyzing:

- ➢ To reset wait statistics: exec qpi.snapshot_wait_stats
- ➢ To read wait statistics (sorted by wait time): select * from qpi.wait_stats

++ recommendation: AUTOMATIC_TUNING

- Always enable AUTOMATIC_TUNING (AT) (FORCE_LAST_GOOD_PLAN) on your MI databases.

Monitor & Tune with Query Store

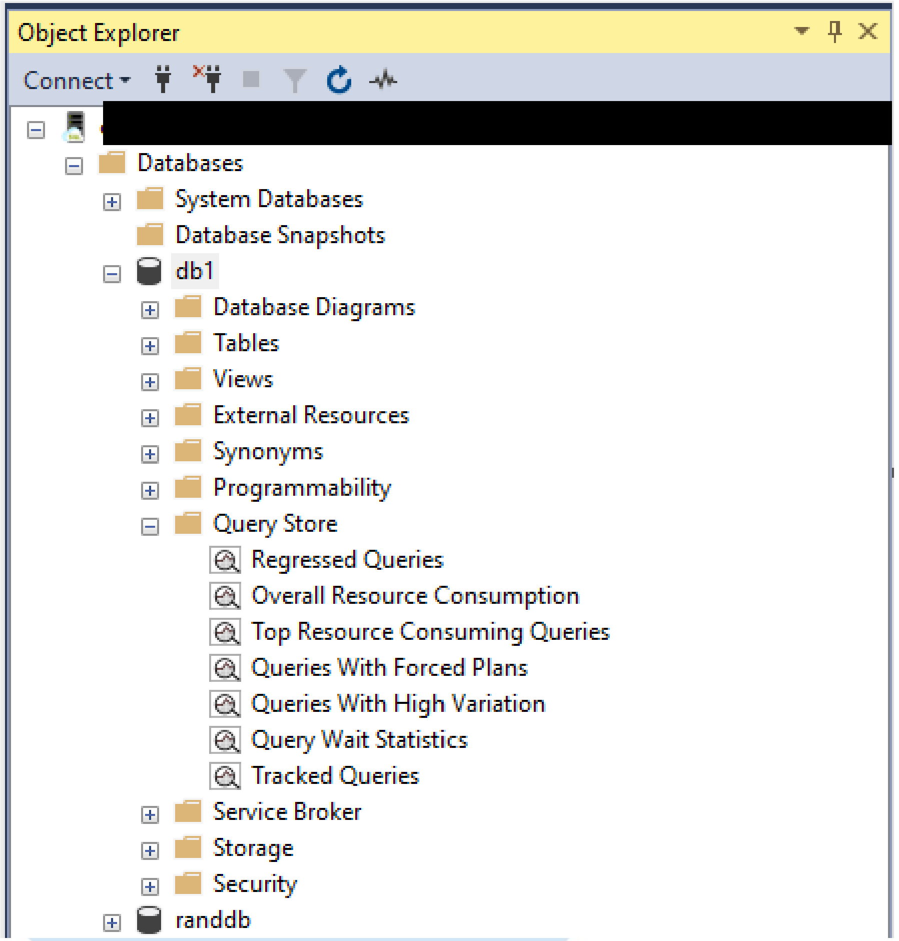
! Query Store cannot be enabled for Master or TempDB databases.

Query Store is enabled by default for new Azure SQL Managed Instance databases.

```
ALTER DATABASE <database_name>
SET QUERY_STORE = ON (OPERATION_MODE = READ_WRITE);
```

Ex: Wait stats

```
ALTER DATABASE <database_name>
SET QUERY_STORE = ON ( WAIT_STATS_CAPTURE_MODE = ON );
```



Visit: Monitor performance by using the Query Store - SQL Server | Microsoft Learn

Best practices for monitoring workloads with Query Store - SQL Server | Microsoft Learn

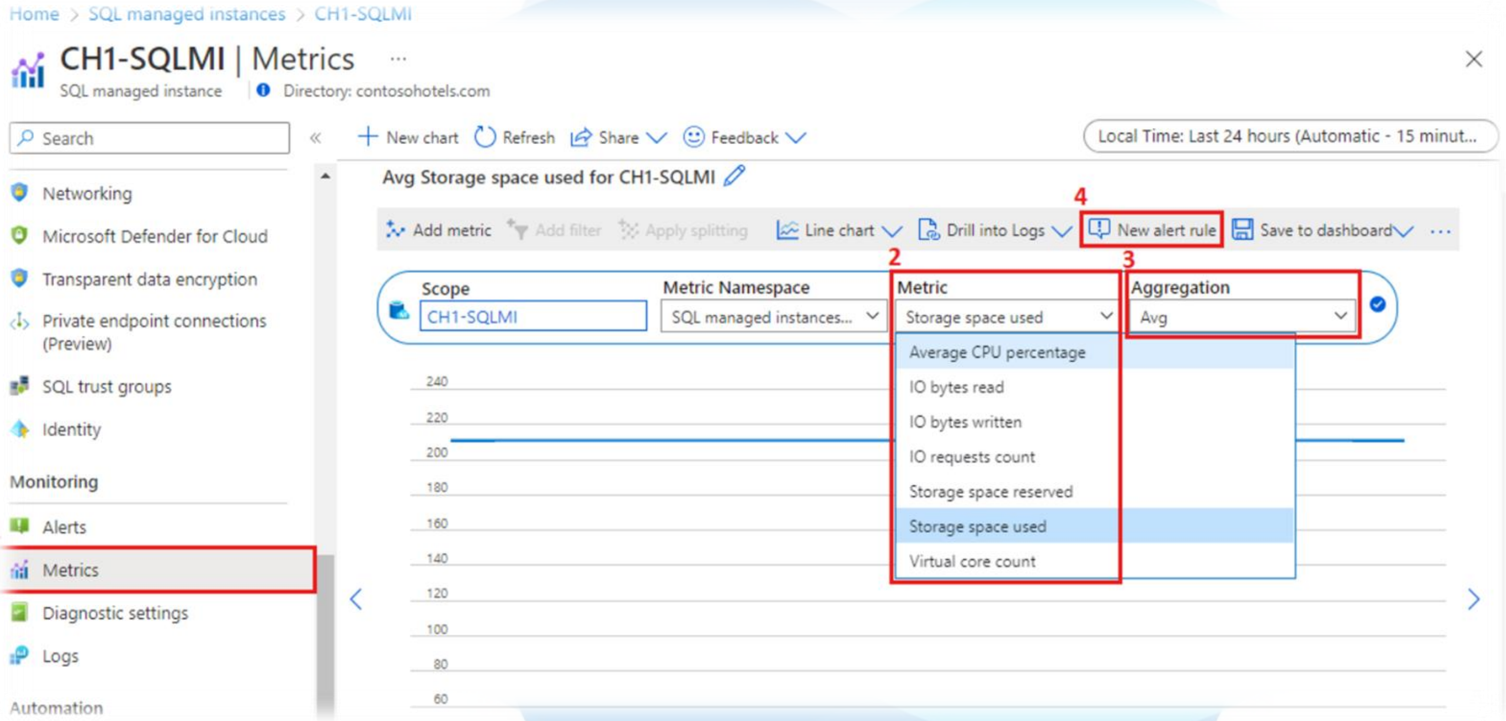
Tune performance with the Query Store - SQL Server | Microsoft Learn

Monitor & Tune with Azure Monitor: Azure Portal

Alerts can do the following when triggered:

- •Send email notifications to the service administrator and co-administrators
- •Send email to additional emails that you specify.
- •Call a phone number with voice prompt
- •Send text message to a phone number
- •Call a webhook
- •Call Azure Function
- •Call Azure runbook
- •Call an external ticketing ITSM compatible system

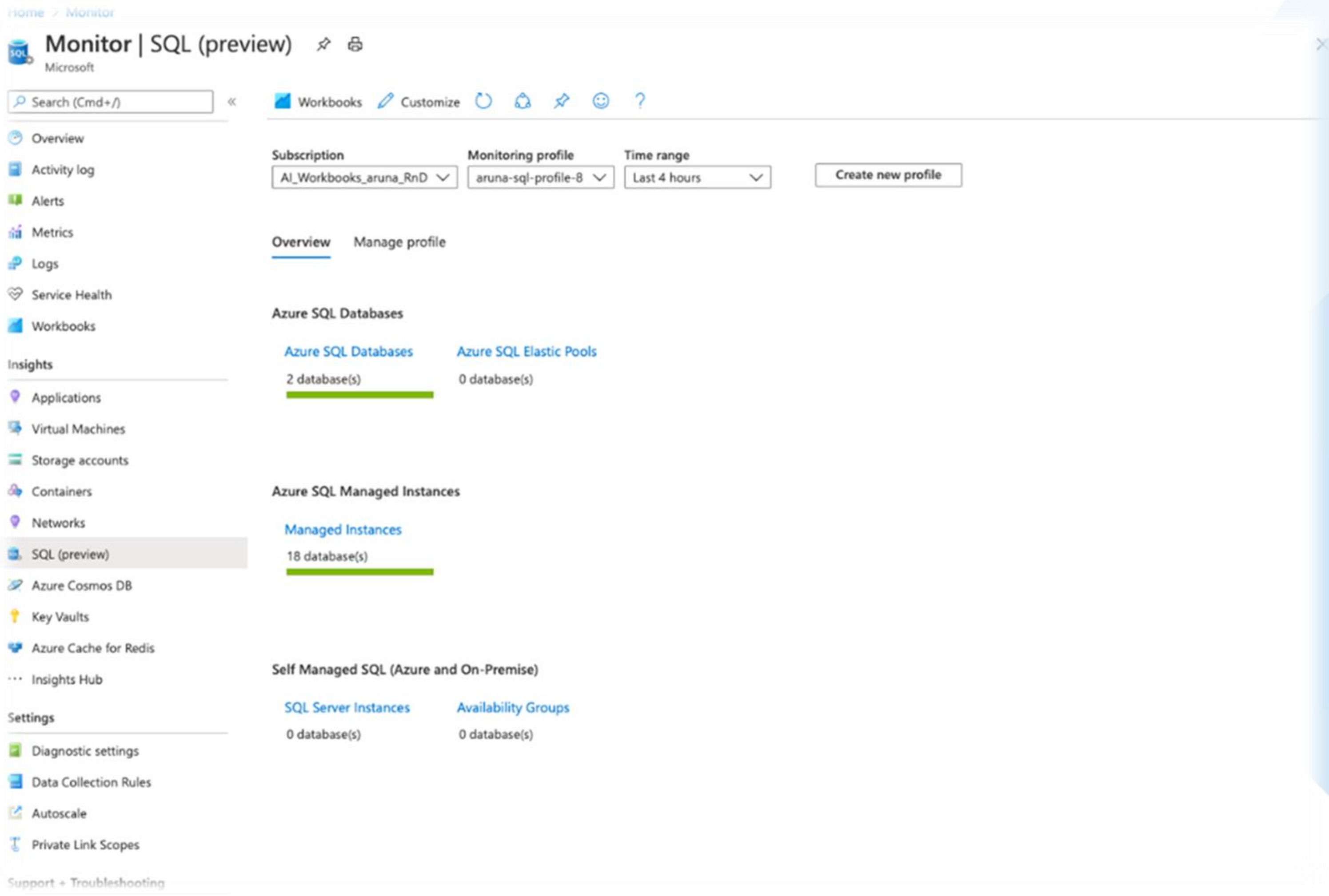
How to set up alerts for databases in Azure SQL MI using the Azure portal:



Scope: CH1-SQLMI
Metric Namespace: SQL managed instances...
Metric: Storage space used
Aggregation: Avg

Visit: [Monitoring Azure SQL Managed Instance with Azure Monitor - Azure SQL Managed Instance | Microsoft Learn]

Monitor & Tune with Azure Monitor: SQL Insights (preview)



Home > Monitor

Monitor | SQL (preview)

Microsoft

Subscription: AI_Workbooks_aruna_RnD

Monitoring profile: aruna-sql-profile-8

Time range: Last 4 hours

Azure SQL Databases

Azure SQL Databases: 2 database(s)

Azure SQL Elastic Pools: 0 database(s)

Azure SQL Managed Instances

Managed Instances: 18 database(s)

Self Managed SQL (Azure and On-Premise)

SQL Server Instances: 0 database(s)

Availability Groups: 0 database(s)

- Monitors any product in the Azure SQL family.
- Uses dynamic management views to expose the data that you need to monitor health, diagnose problems, and tune performance.
- Performs all monitoring remotely.

Visit: Enable SQL Insights (preview) and Troubleshoot SQL Insights (preview)

Key causes of performance differences between SQL MI and SQL Server on-premises

- **Simple or bulk recovery model**
  - select name, recovery_model_desc from sys.databases
  - Indicator: DML transaction processing is slower

- **Resource governance and HA configuration**
  - Indicator: higher INSTANCE_LOG_GOVERNOR wait statistics
  - Resource governance constraints might slow down operations such as bulk load or index rebuild because these operations require higher log rates
  - In addition, the secondary replicas in Business Critical tier instances might slow down the primary database if they can't catch-up the changes and apply them, so you might see additional HADR_DATABASE_FLOW_CONTROL or HADR_THROTTLE_LOG_RATE_SEND_RECV wait statistics.
- **Automated backup schedule**
  - select r.command, query = a.text, start_time, percent_complete, eta = dateadd(second,estimated_completion_time/1000, getdate()) from sys.dm_exec_requests r cross apply sys.dm_exec_sql_text(r.sql_handle) a where r.command IN ('BACKUP DATABASE','BACKUP LOG')
- **Connection and App to Database proximity**
- **Transparent data encryption (TDE)**
  - SQL MI databases are encrypted by default using TDE: select name, is_encrypted from sys.databases
- **Database engine settings** (database compatibility levels, trace flags, system configurations, etc.)
- **Different TempDB configuration**
- **Hardware and environment specification** (Number of cores, Amount of memory including memory/core ratio, IO characteristics,Local or remote storage types, etc.)

Part IV:

Q&A

☺

Reach out to us! Share your feedback and learn more: aka.ms/contactSQLMI

Nevena Nikolic
nnikolic@microsoft.com

Five ways to improve data security on Azure SQL Managed Instance

Microsoft | intel

Thursday June 22, 2023 at 10am PT | 1pm ET | 5pm UTC

When you migrate your data onto Azure SQL Managed Instance, it is secured by multiple layers of protection that go beyond access management to include network security, cluster security, intelligent threat protection, and more – all the way down to the silicon with Intel® Xeon® Scalable processors.

Join this webinar to learn what it means to secure data within a managed service like Azure SQL Managed Instance, including:

- ✓ Setup network isolation with native virtual network support.
- ✓ Securely manage resources within your Azure subscription and mitigate risks of data exfiltration.
- ✓ Enable Windows Authentication for your legacy applications that may not have built-in Azure AD support.
- ✓ Activate threat protection against potential injection attacks and more.
- ✓ Protect your data both in-transit and at rest.

Zoran Rilak

Product Manager
Microsoft

Zoran has been immersed in software engineering since the age of 13. In his previous roles he built and oversaw a data logistics and abstraction layer for a large bioinformatics company, drove data acquisition and mobility at petabyte scale, and helped develop and launch a major new product integrating diverse datasets for scientific and medical research and discovery.

At Microsoft, Zoran focuses on the networking and connectivity aspects of Azure SQL Managed Instance.

Register now: aka.ms/AzureSQLMIWebinar

Microsoft

intel®

Useful links and tips:

- Keep performance stability during the upgrade to newer SQL Server version.
- Monitor CPU usage on your SQL Server instance and check how much compute power (CPU cores) you currently use (using Dynamic Management Views, SQL Server Management Studio, or other monitoring tools).
- Keep in mind that CPU characteristics might need to be scaled to match the characteristics of the VM on which the Managed Instance is installed.
- Check the amount of available memory on your SQL Server instance and choose the service tier that has a matching amount of memory.
- Measure page-life expectancy on your SQL Server instance to determine do you need additional memory.
- Measure IO latency of the file subsystem to choose between service tiers.

Call to action: Get Started Today

aka.ms/SQL_MI_get_started_today