



# Monitoring and Tuning Azure SQL Database

Module 6

## Learning Units covered in this Module

- Lesson 1: Monitoring and Troubleshooting Azure SQL Database
- Lesson 2: Configure Alerts through Azure Portal
- Lesson 3: Monitoring Query Performance using Query Performance Insight
- Lesson 4: Azure SQL Database Tuning using Automatic Tuning
- Lesson 5: Monitoring Azure SQL Database Performance using Database Watcher

# Lesson 1: Monitoring and Troubleshooting Azure SQL Database

# Objectives

After completing this learning, you will be able to:

- Know the various options to monitor and troubleshoot the Azure SQL Database.



# Common Issues on Azure SQL Database

Monitoring for Azure SQL Database is scoped at database level.

Here is list of most faced issues:

Database  
Connectivity

High DTU  
Percentage

Query Timeouts

Deadlocks

Database  
Storage  
consumption

Slow Queries

# Tools to Monitor & Troubleshoot Issues

Query Performance  
Insight

Automatic Tuning

Database Watcher

Dynamic Management  
Views (DMVs)

Azure Database Portal  
Dashboard

Questions?



## Lesson 2: Configure Alerts through Azure Portal



# Objectives

After completing this learning, you will be able to:

- Configure alerts using Azure Management Portal.



# Purpose of Alerts for Azure SQL Database

Database alerts can help to proactively trigger various events related to database connectivity, high DTU usage or deadlocks, etc.

It helps to proactively resolve underlying issues to avoid application outages and improve user experience.

# Receiving an alert based on monitoring metrics or events on

## Metric values

- The alert triggers when the value of a specified metric crosses a threshold you assigned in either direction. It triggers when the condition is first met and then when that condition is no longer being met.

## Activity log events

- An alert can trigger on every event, or, only when a certain number of events occur.

# Purpose of Alerts for Azure SQL Database

You can configure an alert to do the following when it triggers:

- Send email notifications to the service administrator and co-administrators.
- Send email to additional emails that you specify.
- Call a webhook

You can configure and get information about alert rules using

- Azure portal
- PowerShell
- command-line interface (CLI).
- Azure Monitor REST API.

# SQL Database alert values

Metric Name	Aggregation Type	Minimum Alert Time Window
CPU percentage	Average	5 minutes
Data IO percentage	Average	5 minutes
Log IO percentage	Average	5 minutes
DTU percentage	Average	5 minutes
Total database size	Maximum	30 minutes
Successful Connections	Total	10 minutes
Failed Connections	Total	10 minutes
Blocked by Firewall	Total	10 minutes
Deadlocks	Total	10 minutes
Database size percentage	Maximum	30 minutes
In-Memory OLTP storage percent(Preview)	Average	5 minutes
Workers percentage	Average	5 minutes
Sessions percent	Average	5 minutes
DTU limit	Average	5 minutes
DTU used	Average	5 minutes

# Demonstration

## Configure Alerts through Azure Portal

- Configure alerts through Azure Portal.



Questions?



## Lesson 3: Monitoring Query Performance using Query Performance Insight



# Objectives

After completing this learning, you will be able to:

- Know how to troubleshoot the performance of your queries by using Query Performance Insight.



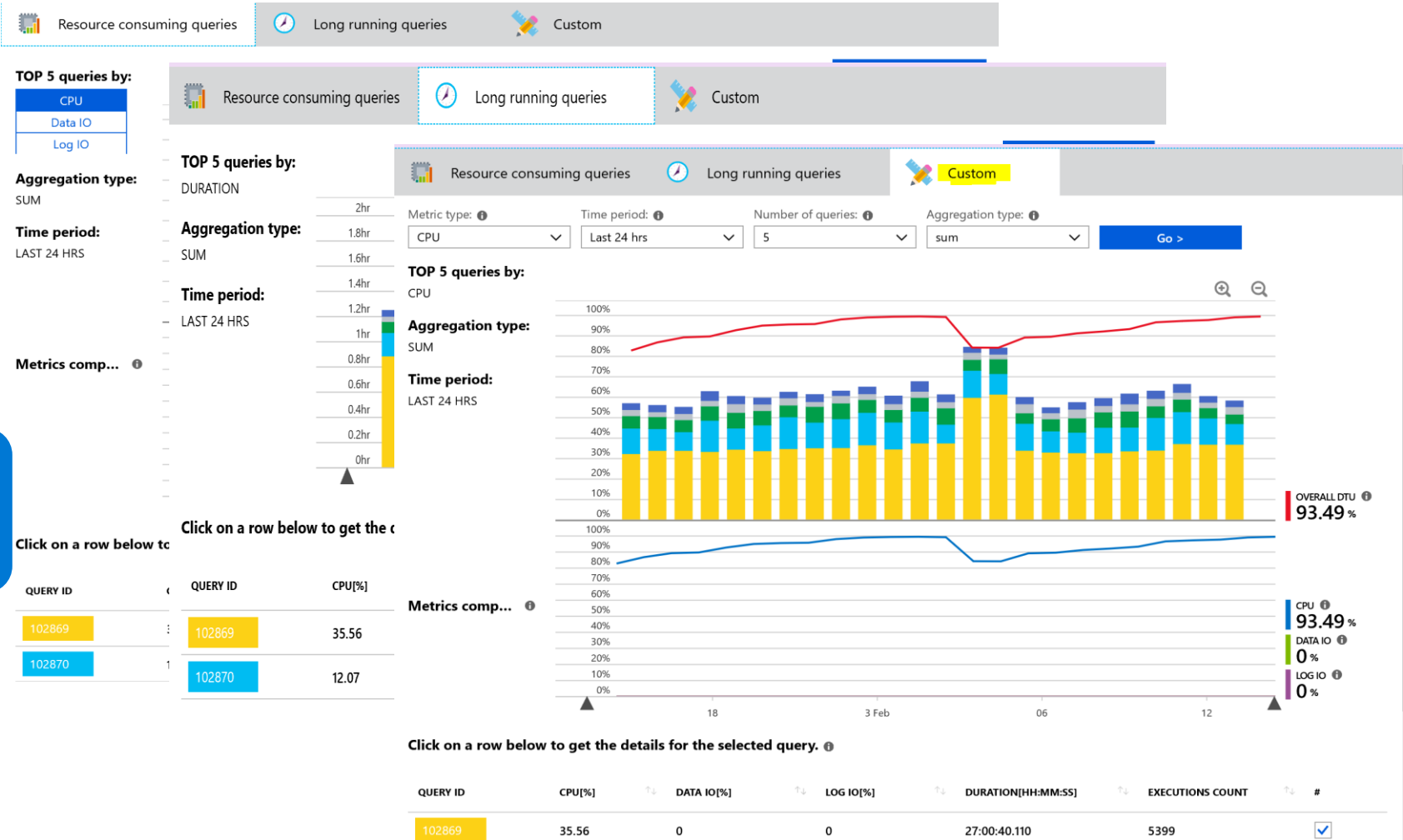
# Query Performance Insight

## Intelligent Performance

- Performance overview
- Performance recommendati...
- Query Performance Insight
- Automatic tuning

## Custom options – Insights based upon custom selection:

- Metric type: Resource consuming queries, Long running queries, and Custom
- Time period: Last 24 hrs, Past Week, Past Month and Custom
- Number of Queries: 5, 10, 20
- Aggregation type: sum, max and avg.
- Log IO utilization %, Duration and Execution count.



# Viewing individual query details

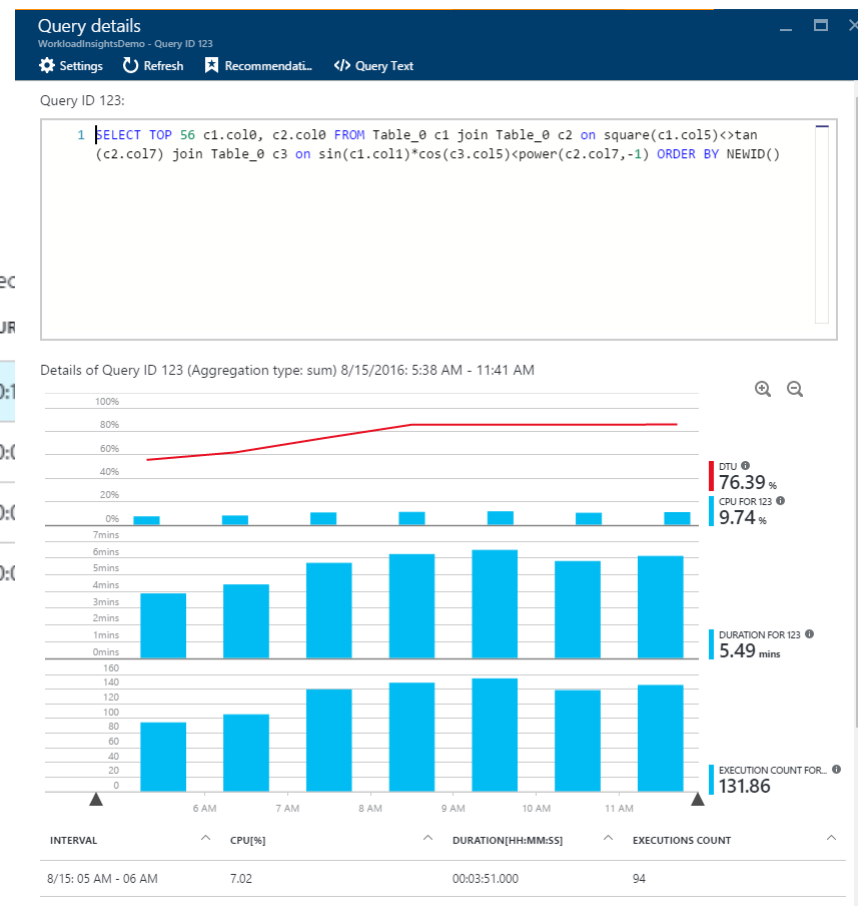
Get details for the individual queries

- CPU Consumption
- Duration
- Execution Count

It does not capture DDL queries

Click on a row below to get the details for the selected query

QUERY ID	CPU[%]	DUR
122	1.27	00:01
123	0.23	00:01
124	0.16	00:01
126	0.09	00:01

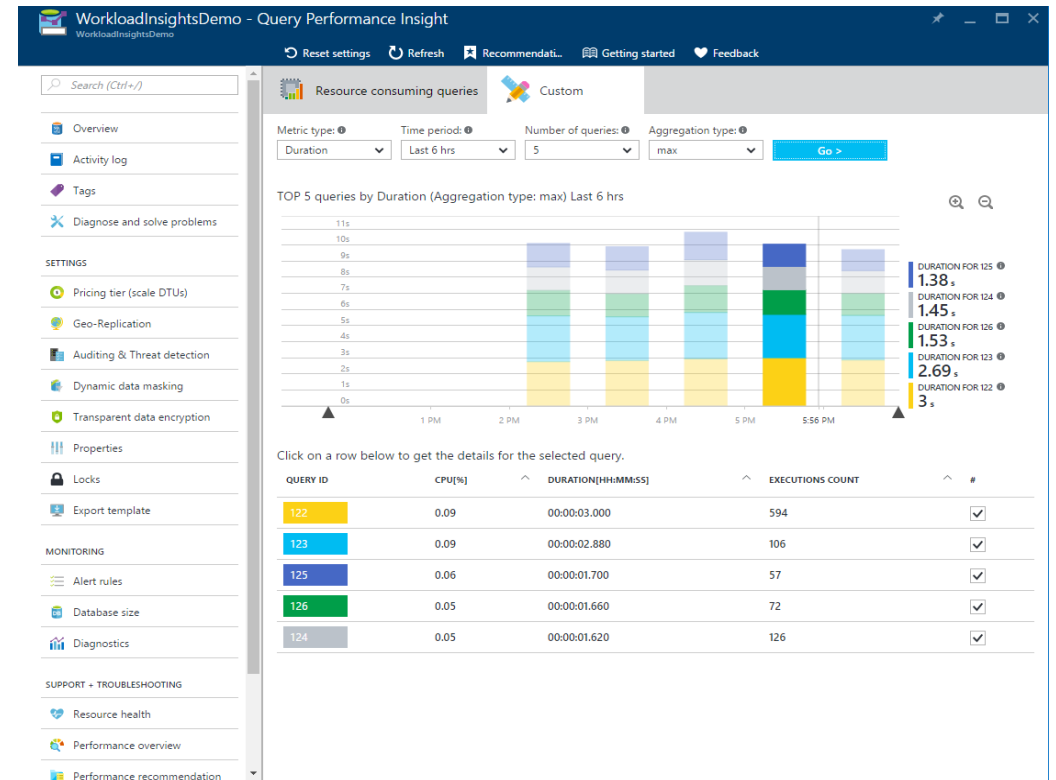


# Review top queries per duration

Duration is one of the metrics showing potential bottleneck

Long-running queries has potential for:

- Longer locks
- Blocking other users
- Limiting scalability

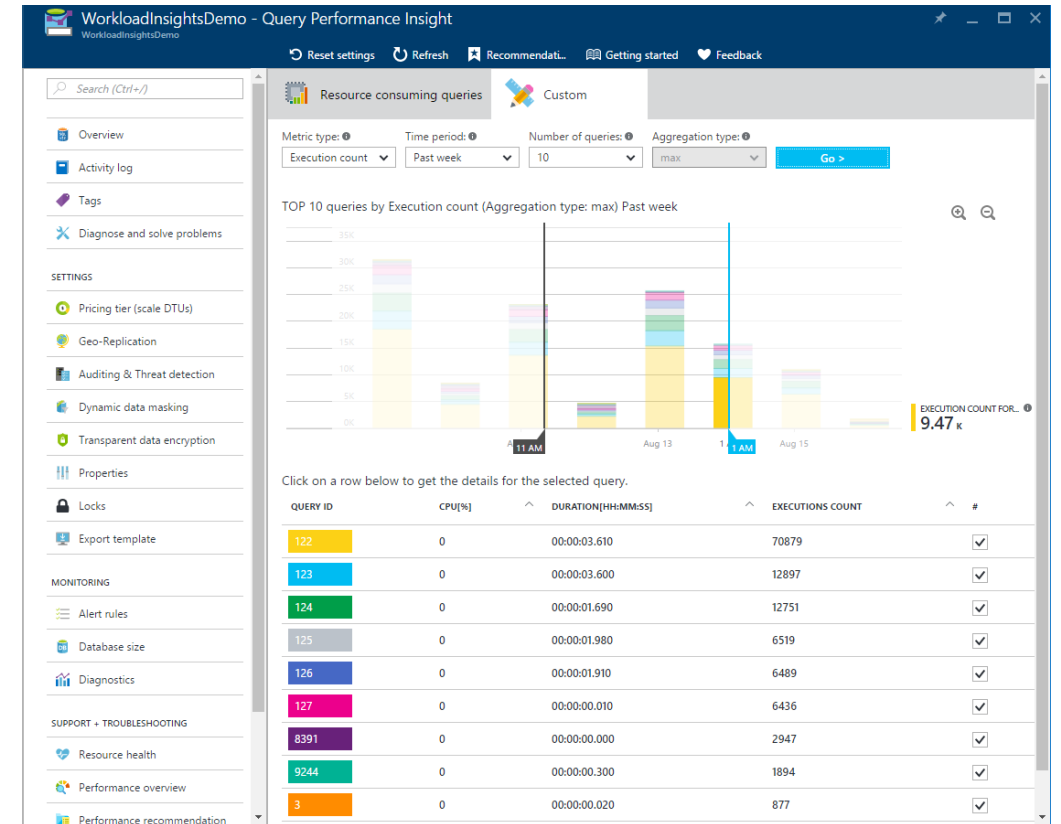


# Review top queries per execution count

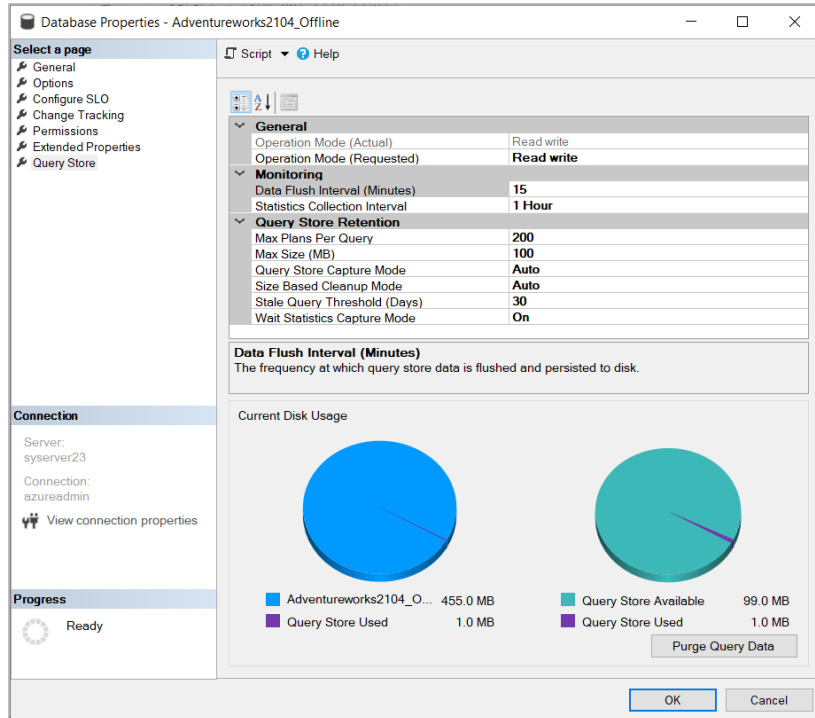
Execution count is one of the metrics showing potential bottleneck

High number of executions has potential for:

- Database performance
- Network latency
- Downstream server latency



# Query Store



## Retention Policy

- Size based – Auto cleanup when near max size.
- Time based – Default 30 days.
- Max Plans Per Query – Default 200.
- Wait Statistics Capture Mode – Default On.

## Capture Policy

- All – Captures all queries.
- Auto – Infrequent queries are ignored.
- None – No queries are captured.
- Custom – Advanced Options

# Demonstration

## Query Performance Insight

- Analyze the Query Performance Insight output.



# Monitoring Query Performance using Query Performance Insights

- Configure the Query Store.
- Analyze the Query Performance Insight.





Questions?



# Knowledge Check

What feature should be enabled on your Azure SQL Database before you can use Query Performance Insight?

How can you view individual query details?

# Lesson 4: Azure SQL Database Tuning using Automatic Tuning

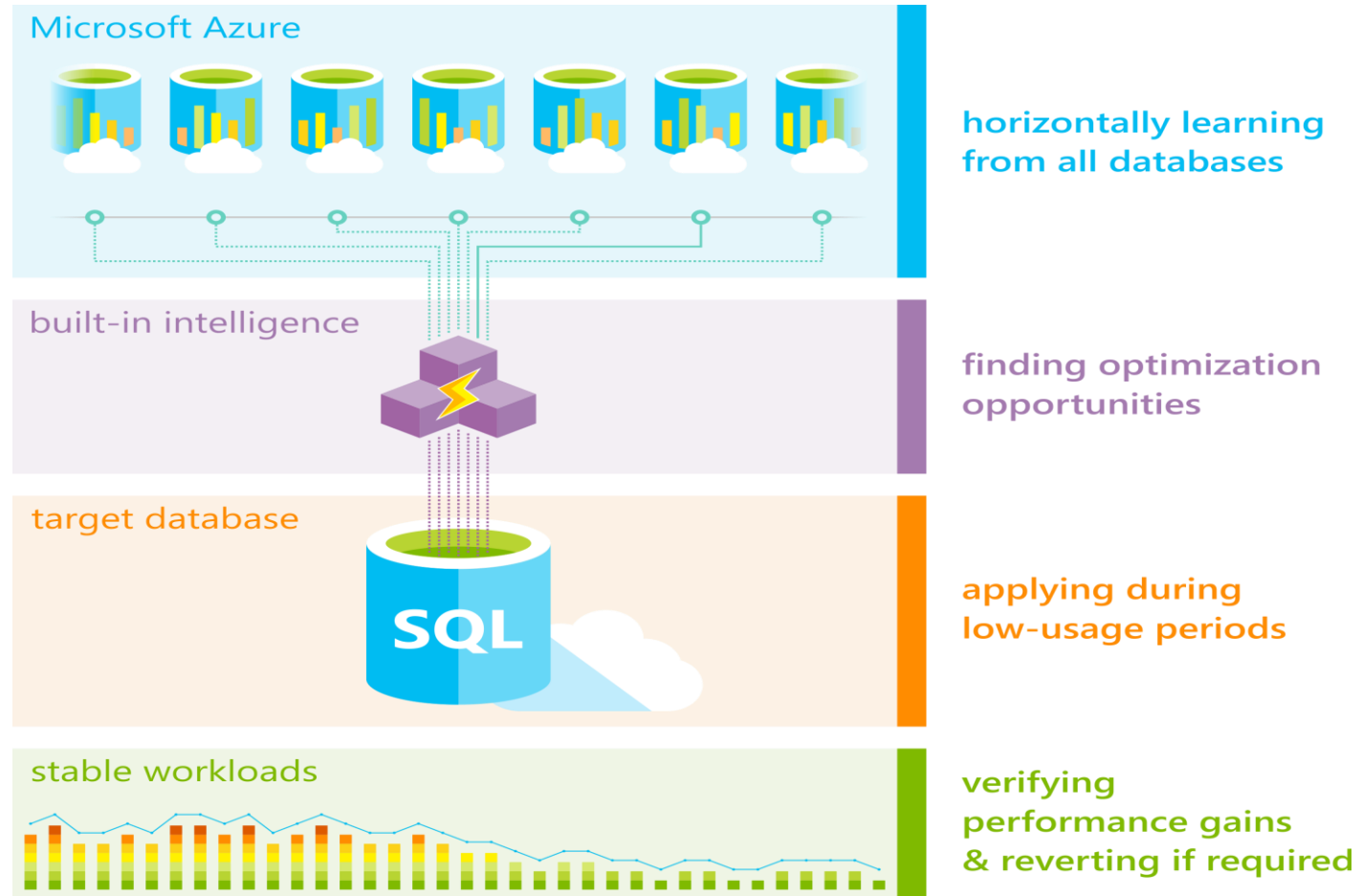
# Objectives

After completing this learning, you will be able to:

- Know how Performance Recommendations can help to improve database performance.




# Automatic Tuning



[Performance recommendations for SQL Database](#)

# Intelligent Performance – Automatic Tuning

### Intelligent Performance




 Performance overview

Inherit from: ⓘ

Server | **Azure defaults** | Don't inherit

ⓘ The database is inheriting automatic tuning configuration

Configure the automatic tuning options ⓘ

OPTION	
	FORCE PLAN
	CREATE INDEX
	DROP INDEX

Estimated impact

**Validation report**

▼ Validation progress ⓘ	Completed
DTU savings (overall) ⓘ	31.75% DTU
DTU savings (affected queries) ⓘ	90.00% DTU
Queries with improved performance ⓘ	12
Queries with regressed performance ⓘ	1

**Force Last Good Plan:**

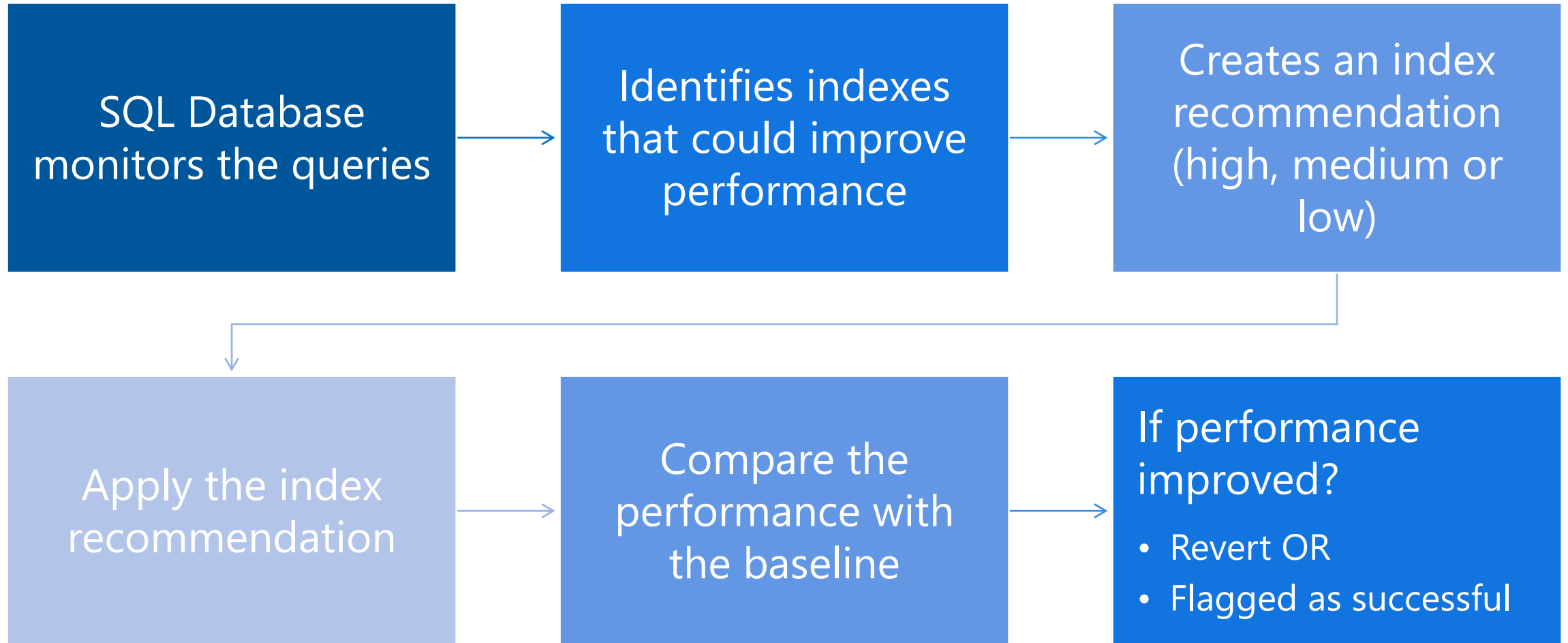
ies due to bad plan with last performance. It tests the change if it improves.

lexes, validates performance improvements and reverts the change if performance degrades.

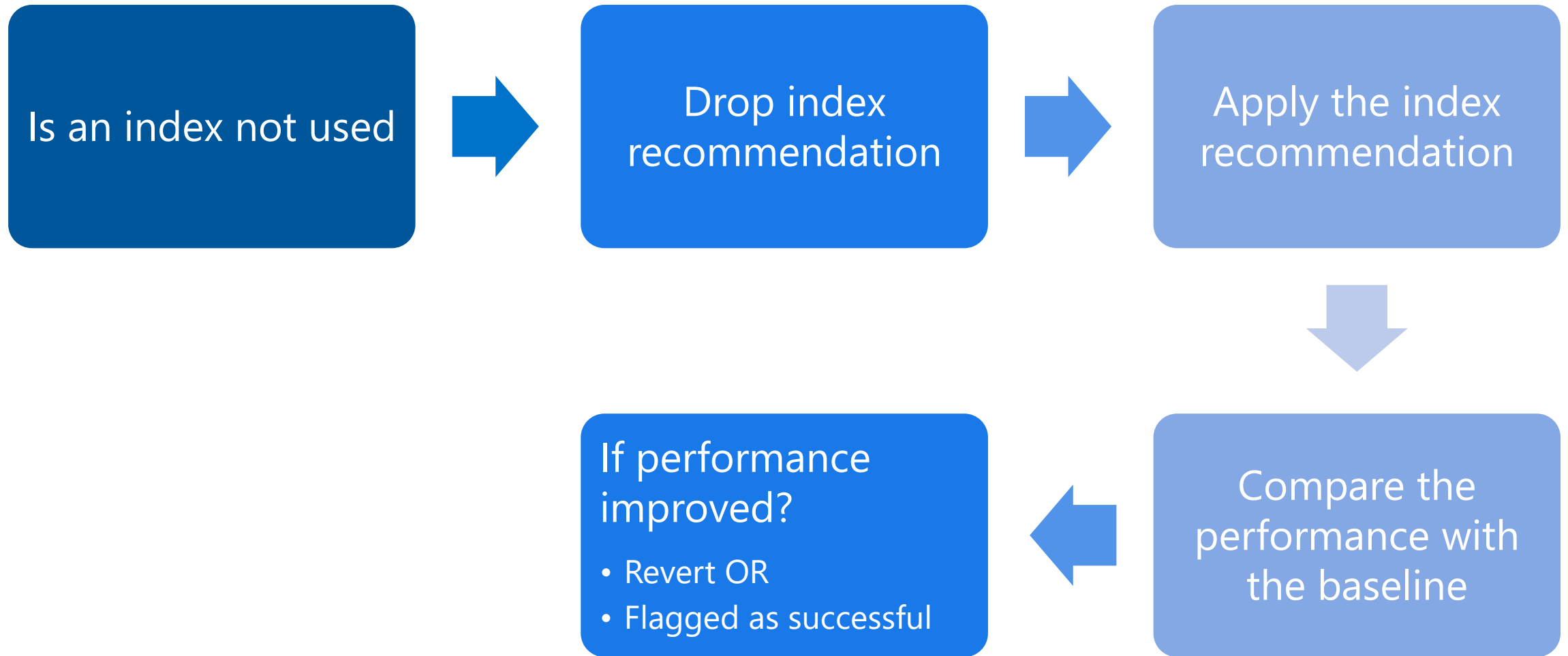
- Identifies and drops unused Indexes, validates performance improvements and reverts the change if performance degrades.

<http://automaticplan correctiondemo.azurewebsites.net/index.html>

# Automatic Tuning – Create Index

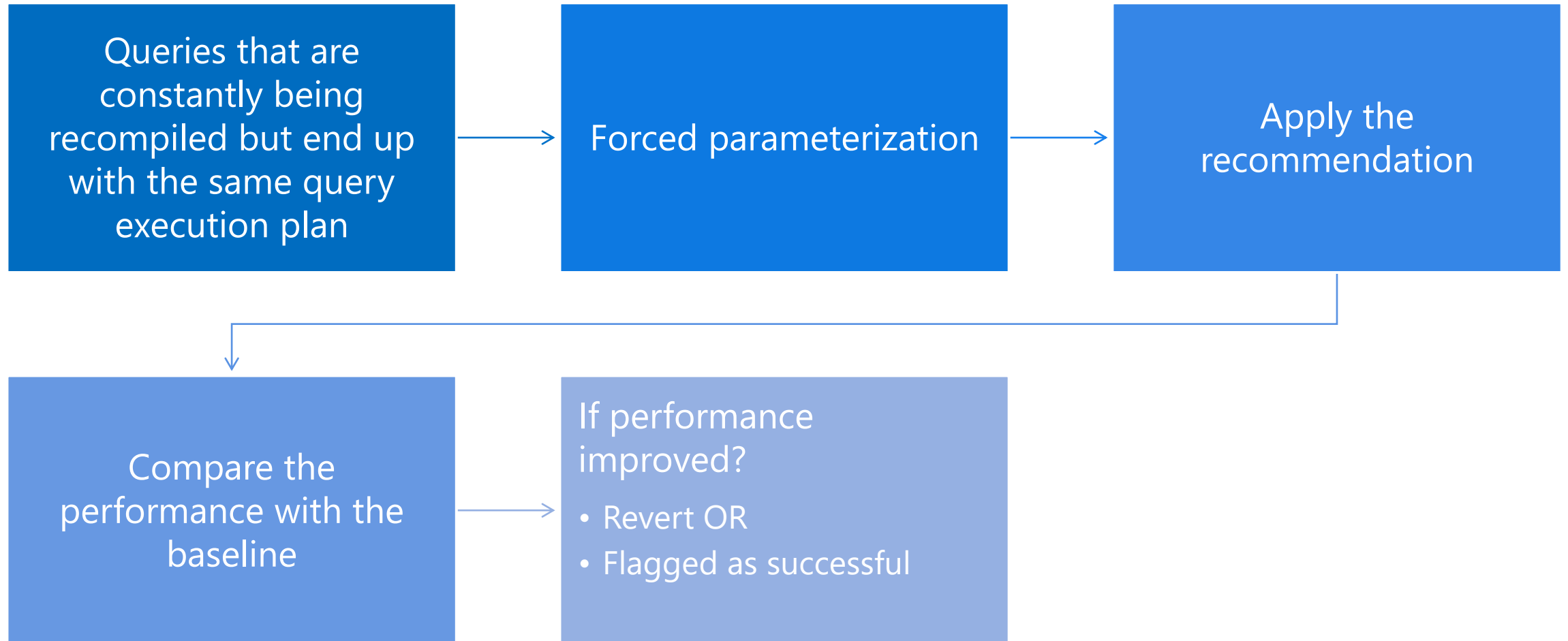


# Automatic Tuning – Drop Index





# Automatic Tuning – Parameterize Queries



Questions?



# Knowledge Check

List three types of recommendations from Automatic Tuning.

What could be a reason to disable the automatic tuning option?

What technology is used for Automatic Tuning?

# Lesson 5: Using SSMS Built-In Reports

# Objectives

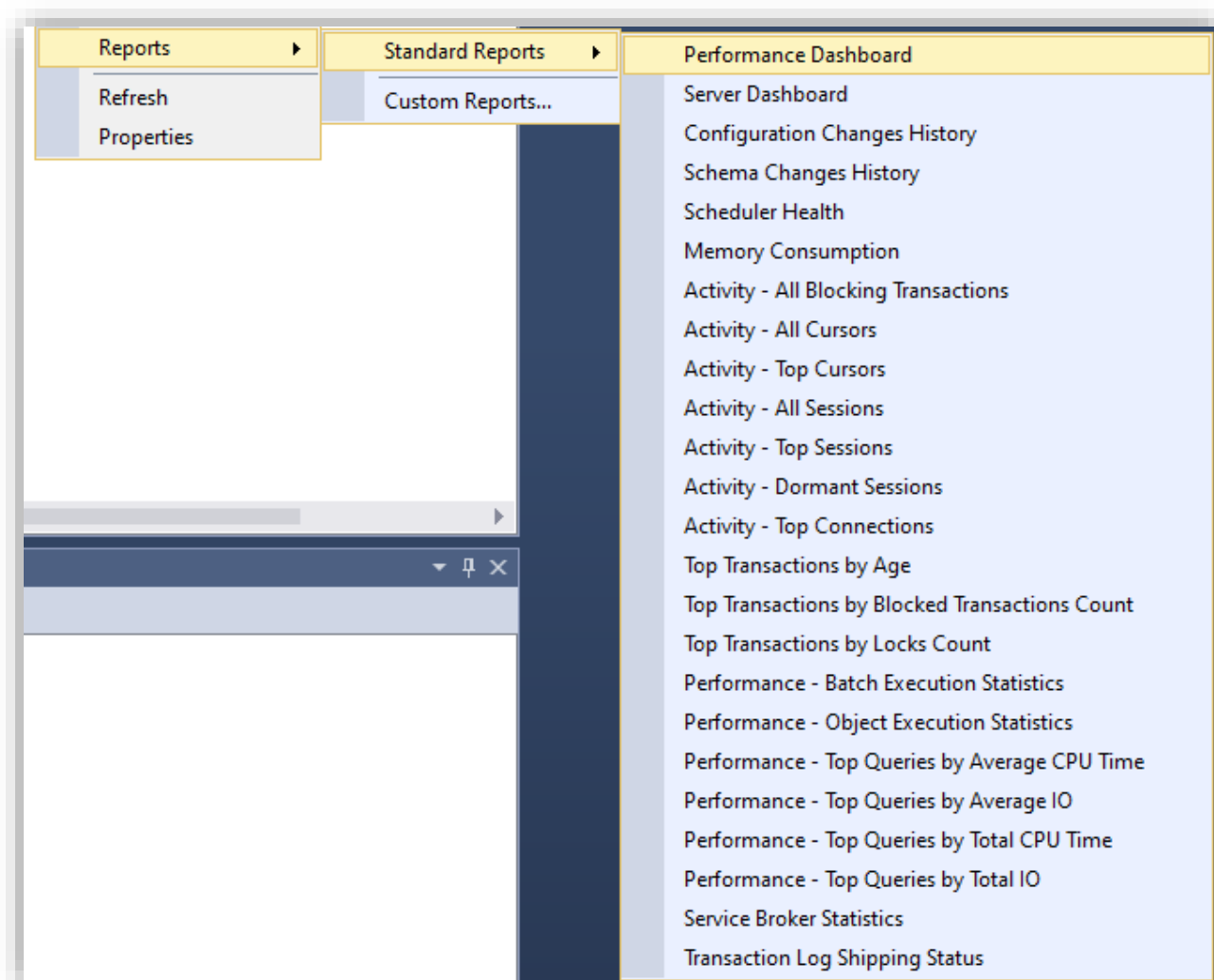
After completing this learning, you will be able to:

- Understand Built-In Instance Reports and Database Reports



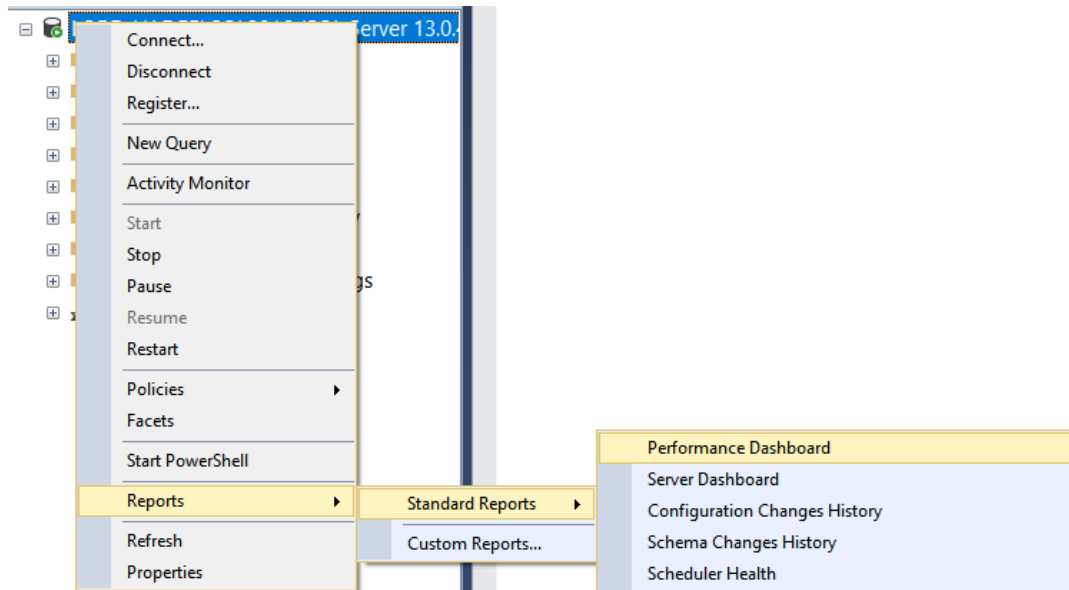
# Instance Reports

# Built-in Instance Reports



# Performance Dashboard

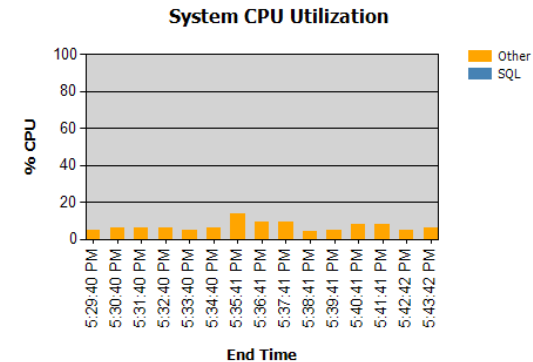
- To view the Performance Dashboard, right-click on the SQL Server instance name in Object Explorer, select **Reports, Standard Reports**, and click on **Performance Dashboard**.




## Microsoft SQL Server Performance Dashboard

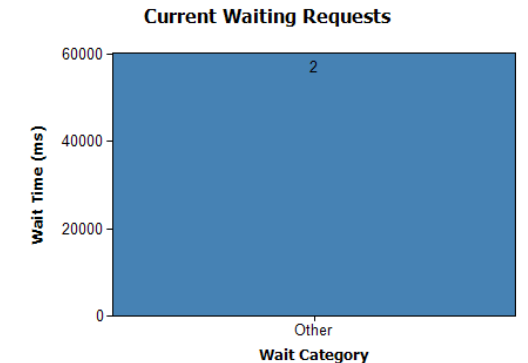
Report Local Time: 11/5/2020 5:44:05 PM

(12.0.2000.8 - SQL Azure)



Current Activity		
	User Requests	User Sessions
Count	3	13
Elapsed Time (ms)	1377628468	2736
CPU Time (ms)	886(0.00%)	188(6.87%)
Wait Time (ms)	1377627582(100.00%)	2548(93.13%)
Cache Hit Ratio	46.218%	88.058%

 System performance may be degraded because of excessive waits happening on the server. Click on a Wait Category data point in the chart below to investigate further.

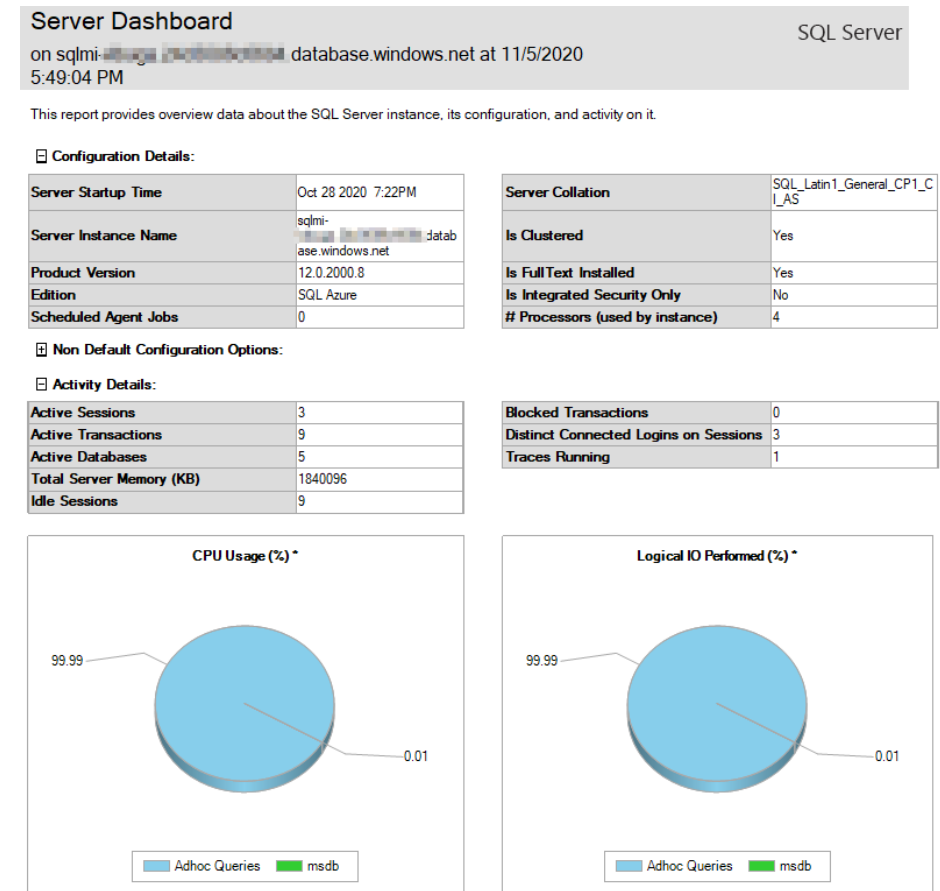


Historical Information	
<a href="#">Waits</a>	<a href="#">IO Statistics</a>
<a href="#">Latches</a>	
Expensive Queries	
<a href="#">By CPU</a>	<a href="#">By Duration</a>
<a href="#">By Logical Reads</a>	<a href="#">By Physical Reads</a>
<a href="#">By Logical Writes</a>	<a href="#">By CLR Time</a>
Miscellaneous Information	
<a href="#">Active Traces</a>	1
<a href="#">Active Xevent Sessions</a>	3
<a href="#">Databases</a>	5



# Server Dashboard

- Report provides overview data about SQL Server Instance, configuration and activity on it.
  - Configuration Details
    - SQL Startup Time, Instance Name, Product Version
    - SQL Collation
    - Is Clustered, Is Integrated Security Only
    - # Processors
  - Non-Default Configuration Options
    - Traceflag, Run Value, Default Value
  - Activity Details
    - Active Sessions, Transaction, Databases
    - Total Server Memory, Idle Sessions
    - Blocked Transactions



\* : "CPU Usage" and "IO Performed" charts show the cumulative share of all objects by databases.

# Configuration Change History

## Global Trace Flags – Configuration Options

- Provides a history of instance configuration and trace flag changes
- Reads history from **Default Trace**

### Configuration Changes History

on [redacted].database.windows.net at 11/19/2020 1:21:24 PM

SQL Server

This report provides a history of all sp\_configure and Trace Flag changes recorded by the Default Trace.

**Configuration Changes History (Since 11/19/2020 1:21:10 PM).**

Shows changes in server configuration and flags.

Configuration Option	Old Value	New Value	Time	User
Trace Flag (11024, -1)	--	on	11/19/2020 1:21:10 PM	AzureAdmin

# Schema Change History

## DDL Statement Commits

- Provides a history of schema changes

Schema Changes Hi...abase.windows.net X Schema Changes Hi...abase.windows.net

Schema Changes History

SQL Server

on [redacted].database.windows.net at 11/19/2020 4:52:30 PM

This report provides a history of all committed DDL statement executions recorded by the default trace.

Schema Change History (Since 11/19/2020 4:51:08 PM ).

Shows changes made in the schema of the objects by DDL operations.

Database Name	Object Name	Type	DDL Operation	Time	Login Name
6cc8648e-d451-42ca-a103-dbcf85ab3dcb	Table_1	User Defined Table	ALTER	11/19/2020 4:52:21 PM	AzureAdmin
6cc8648e-d451-42ca-a103-dbcf85ab3dcb	Table_1	User Defined Table	CREATE	11/19/2020 4:52:21 PM	AzureAdmin

# Performance Reports

## Batch Execution Statistics

- Provides execution history data for all cached batch plans.

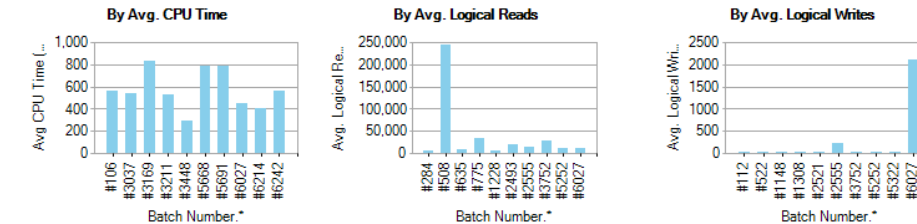
### Performance - Batch Execution Statistics

SQL Server

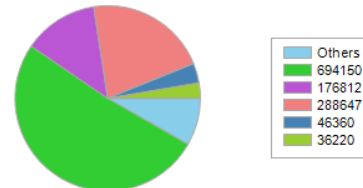
on [192.168.1.104](#).database.windows.net at 11/19/2020  
5:33:18 PM

This report provides detailed historical execution data for all currently cached batch plans. This execution data is aggregated over the time during which the plan has been in the cache.

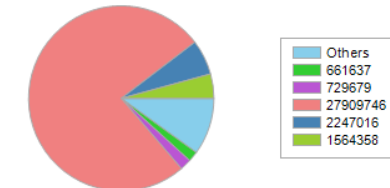
#### Top Batches



#### Total CPU Time (%) By Batches\*



#### Total Logical IO (%) By Batches\*



\* See the "Batch Number" column in the table below for the batch numbers reported in the charts.

#### SQL Batches

Shows statement wise execution statistics for all the objects.

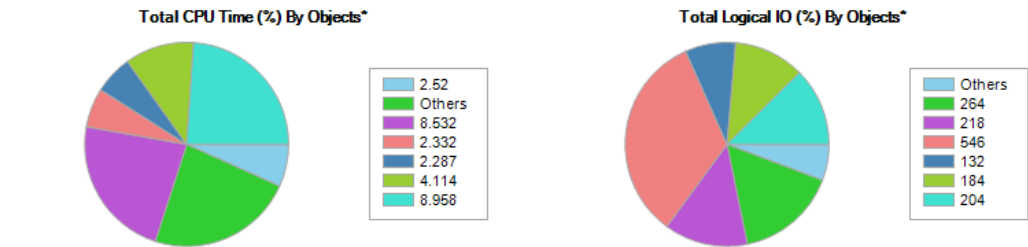
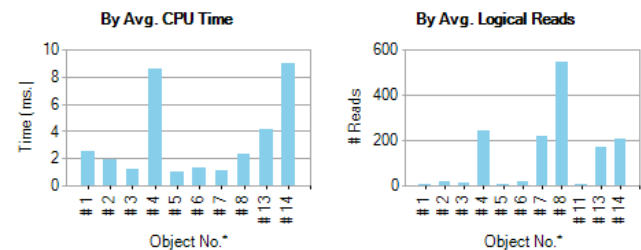
Batch Number	First SQL Statement of Batch	Avg. CPU Time (ms.)	# Avg. Logical Reads	# Avg. Logical Writes
1	SELECT *, 861 FROM [AdventureWorks].[Person].[Person] WHERE [BusinessEntityID] = @BusinessEntityI	0.19	3.00	0.00
2	SELECT *, 821 FROM [AdventureWorks].[Person].[Person] WHERE [BusinessEntityID] = @BusinessEntityI	0.18	3.00	0.00

# Performance Reports

## Object Execution Statistics

- Provides execution history data for all cached plans.

Top Executable Objects



All Executable Objects

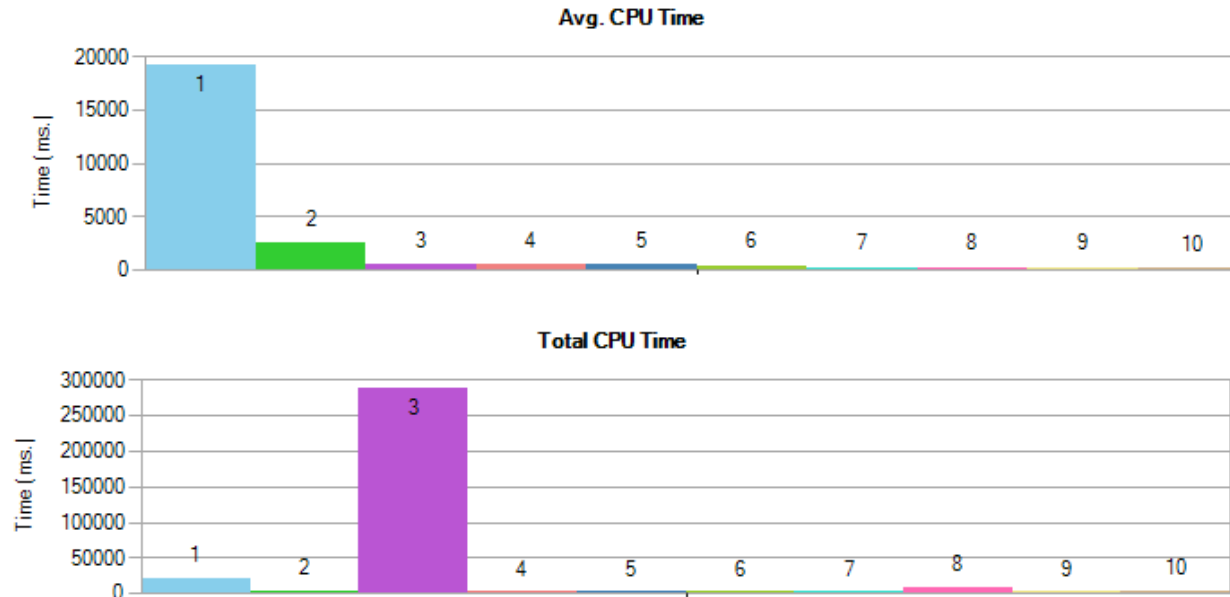
Shows statement wise execution statistics for all the executable objects.

Object No.*	Database Name	Object Name	Object Type	Avg. CPU Time (ms.)	Total CPU Time (%)	# Avg. Logical Reads	# Avg. Logical Writes	# Avg. Logical IO	Total Logical IO (%)
1	msdb	managed_backup fn_backup_db_config	SQL Table-valued-Function	2.52	6.73	4.00	0.00	4.00	0.24
SQL Statement				# Executions (With Last Plan)	# Plans Generated	Avg. CPU Time (ms.)	# Avg. Logical Reads	# Avg. Logical Writes	# Avg. Logical IO
INSERT INTO @t SELECT aamd.db_name, aamd.db_guid, CASE WHEN aamd.group_db_guid IS NULL THEN CONVERT(BIT, False) END AS group_db_guid				1	1	2.52	4.00	0.00	4.00

# Performance Reports

## Top Queries by Average CPU Time

- Provides top queries by average CPU time for all cached plans.

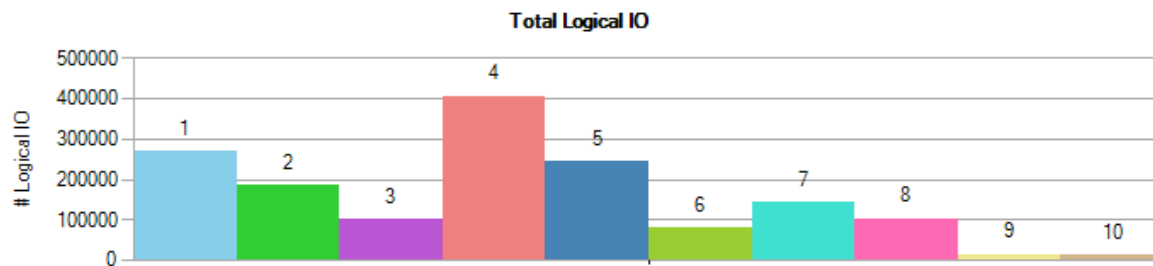
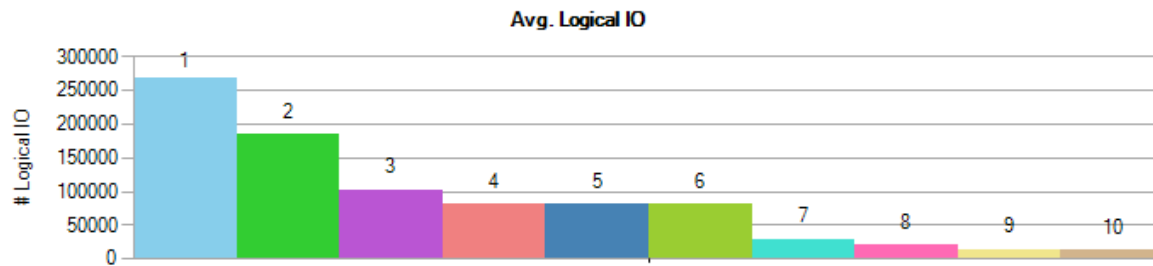


Query No.	Query Text	Database Name	Object ID	Avg. CPU Time (ms.)
1	<pre>select (s.t_SPRank)%2 as I1 , (dense_rank() over(order by s.t_SPRank,s.row_id))%2 as I2 , s.* , ob.cpu_rank as t_CPURank , ob.read_rank as t_ReadRank , ob.write_rank as t_WriteRank from @sql_handle_convert_table s join @objects ob on (s.t_SPRank = ob.obj_rank</pre>			19,204.53
2	<pre>insert into @sql_handle_convert_table Select sql_handle , sql_handle as chart_display_option , sql_handle as chart_display_optionIO , master.dbo.fn_varbintohexstr(sql_handle) , dense_rank() over (order by s1.sql_handle) as SPRank , dense_rank() over (partition by s1.sql_handle order by s1.statement_start_offset) as SPRank2 , (select top 1 substring(text,(s1.statement_start_offset+2)/2, (case when s1.statement_end_offset = -1 then len(convert (nvarchar(max),text))"2 else s1.statement_end_offset end - s1.statement_start_offset)/2 ) from sys.dm_exec_sql_text (s1.sql_handle)) as [SQL Statement] , execution_count , plan_generation_num , last_execution_time , (total_worker_time+0.0)/execution_count/1000 as [avg_worker_time] , total_worker_time/1000 , last_worker_time/1000 , min_worker_time/1000 , max_worker_time/1000 , (total_logical_reads+0.0)/execution_count) as</pre>			2,552.09

# Performance Reports

## Top Queries by Average IO

- Provides top queries by average IO for all cached plans.



Query No.	Query Text	Database Name	Object ID	# Avg. Logical IO
1	<pre>select (s.t_SPRank)%2 as I1 , (dense_rank() over(order by s.t_SPRank,s.row_id))%2 as I2 , s.* , ob.cpu_rank as t_CPURank , ob.read_rank as t_ReadRank , ob.write_rank as t_WriteRank from @sql_handle_convert_table s join @objects ob on (s.t_SPRank = ob.obj_rank</pre>			268,351.00
2	<pre>select top 10 rank() over(order by (total_worker_time +0.0)/execution_count desc,sql_handle,statement_start_offset ) as row_no , (rank() over(order by (total_worker_time +0.0)/execution_count desc,sql_handle,statement_start_offset ))%2 as I1 , creation_time , last_execution_time , (total_worker_time+0.0)/1000 as total_worker_time , (total_worker_time+0.0)/(execution_count*1000) as [AvgCPUTime] , total_logical_reads as [LogicalReads] , total_logical_writes as [LogicalWrites] , execution_count , total_logical_reads+total_logical_writes as [AggIO] , (total_logical_reads+total_logical_writes)/(execution_count +0.0) as [AvgIO] , case when sql_handle IS NULL</pre>			184,550.00

# Database Reports



# Built-In Database Reports

Reports	Standard Reports	Disk Usage
Rename	Custom Reports...	Data Classification
Delete	All Transactions	Disk Usage by Top Tables
Refresh	Disk Usage	Disk Usage by Table
Properties	All Blocking Transactions	Disk Usage by Partition
		Backup and Restore Events
		All Transactions
		All Blocking Transactions
		Top Transactions by Age
		Top Transactions by Blocked Transactions Count
		Top Transactions by Locks Count
		Resource Locking Statistics by Objects
		Object Execution Statistics
		Database Consistency History
		Transaction Performance Analysis Overview
		Index Usage Statistics
		Index Physical Statistics
		Schema Changes History
		User Statistics

# Demonstration

## Server Reports

- SSMS Built-In Server Reports



# Lesson 6: Monitoring Performance using Database Watcher

# Objectives

After completing this learning, you will be able to:

- Use Database Watcher to monitor database performance.
- Use Database Watcher to monitor any product in the Azure SQL family.



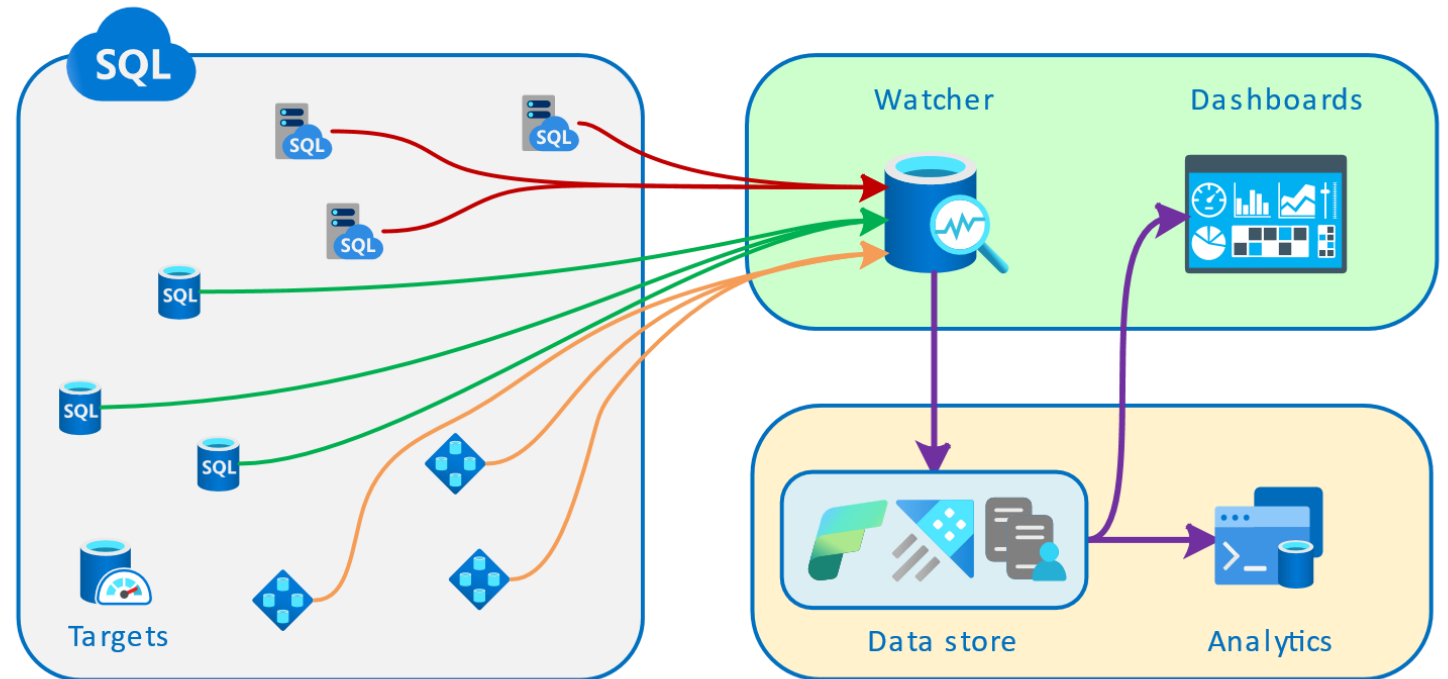
# Introduction to Database Watcher (Preview)

Database watcher is a managed monitoring solution for database services in the Azure SQL family.

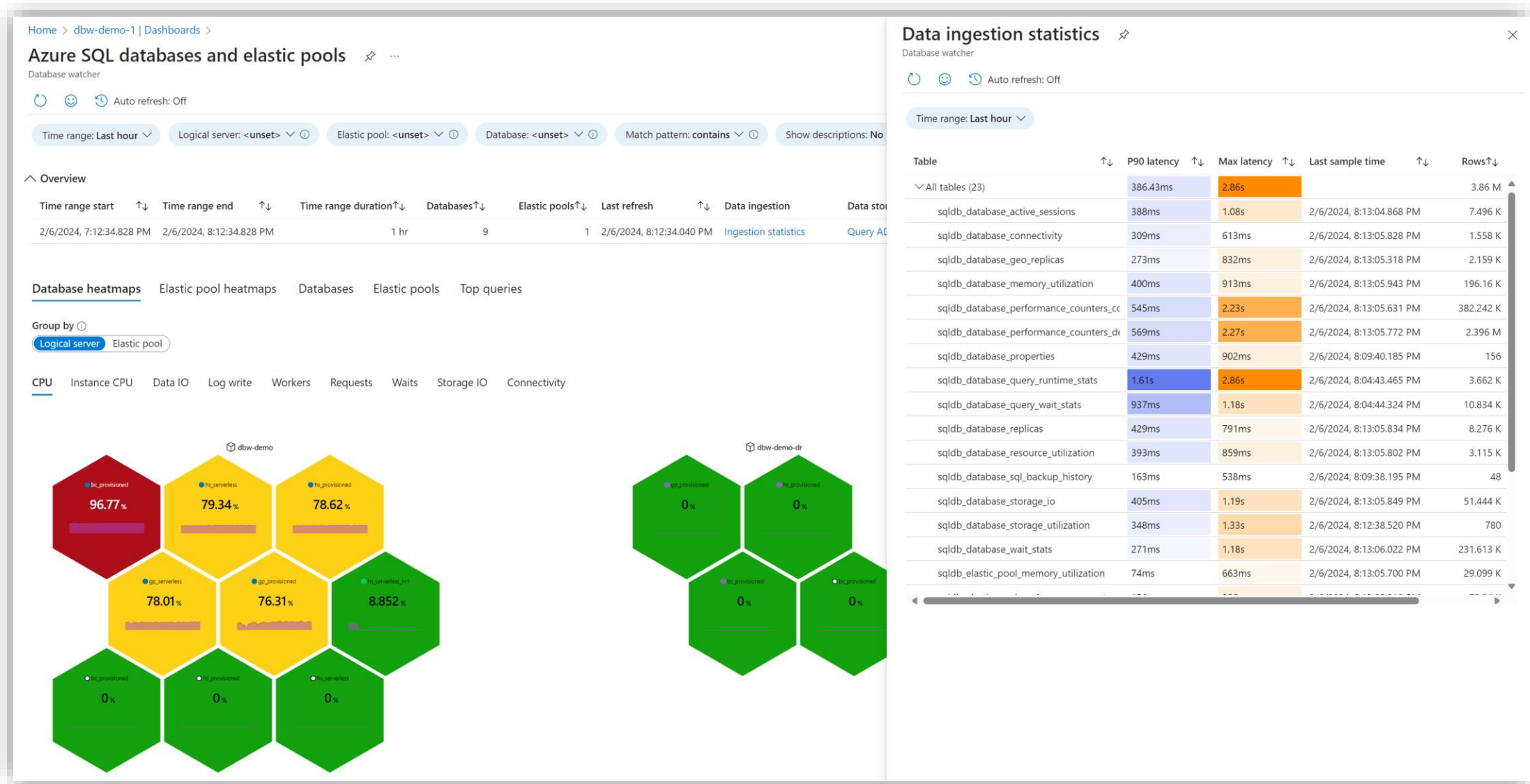
Supports Azure SQL Database, Elastic Pools, and Managed Instances

Collects workload monitoring data to give you a detailed view of database performance, configuration, and health.

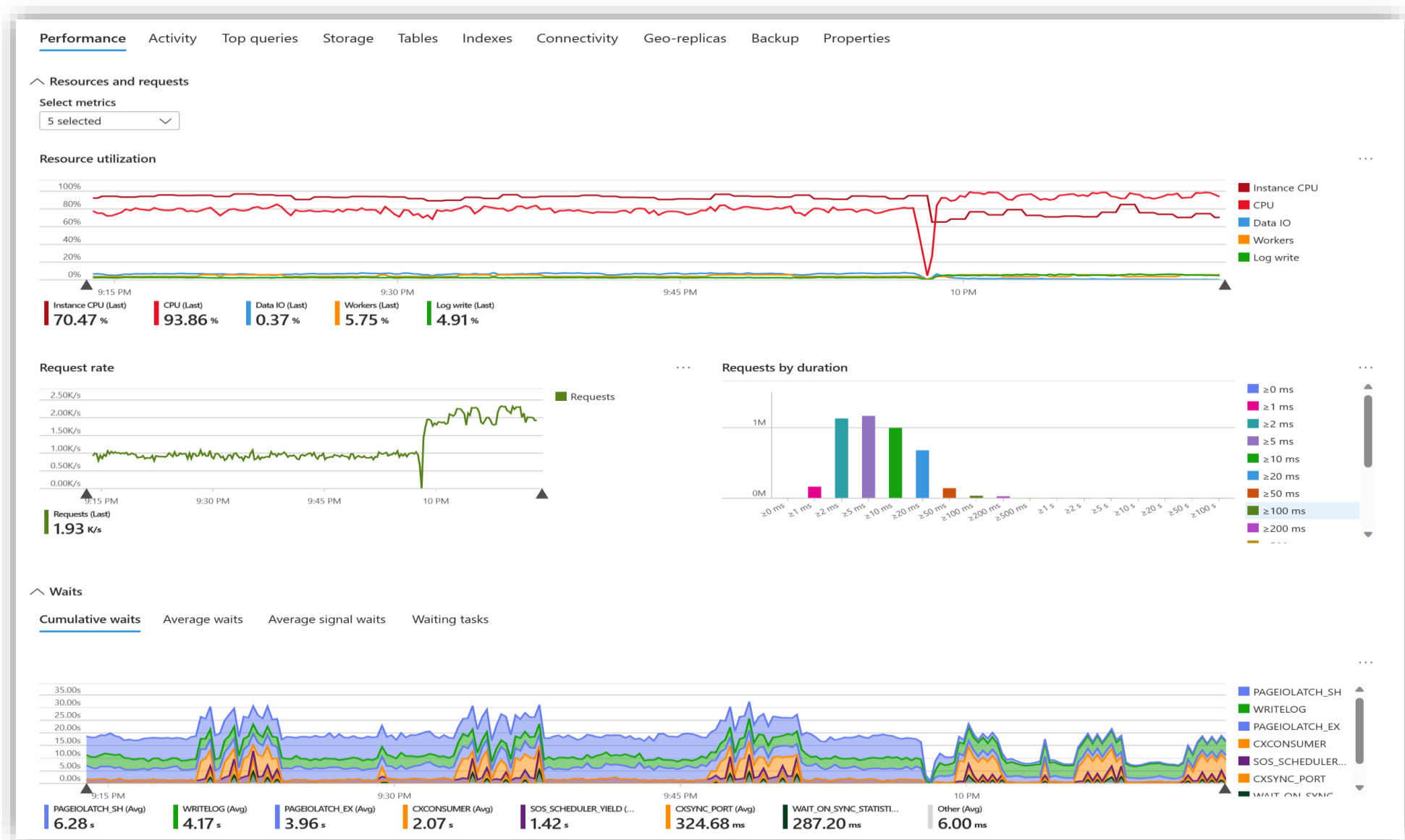
Currently in Preview and only available in limited Azure regions.



# Database Watcher Dashboards



# Database Watcher Dashboards



# Create a Database Watcher Service

Home >

Database watchers

+ Create

Manage view

Refresh

Export to CSV

Filter for any field...

Subscription equals


Showing 0 to 0 of 0 records.

No group

Name ↑↓

Type ↑↓

Resource group ↑↓



No database watchers to

Home > Database watchers >

Create a database watcher

Basics

Identity

Data store

Targets

Review + create

Create a database watcher to monitor your Azure SQL resources in-depth and at scale. Once a database watcher is created, you can enable monitoring for your Azure SQL databases, elastic pools, and managed instances. [Learn more](#)

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \*

Resource group \*

(New) database-watcher-quickstart

Create new

Instance details

Name \*

example-watcher-1

Region \*

(US) East US



# Add Targets to Database Watcher

Home > example-watcher-1

example-watcher-1 | SQL targets ☆ ...

Database watcher

Search ◇ << **+ Add** 🗑 Delete ↻ Refresh

Overview

Activity log

Access control (IAM)

Resource visualizer

> Favorites

> Monitoring

✓ Configuration

🔑 Identity

🔗 Managed private endpoints

**SQL targets**

🔗 Data store

> Automation

**Resource**

📘 No targets found

To grant this watcher access to collect metrics on the Azure SQL logical servers contained in this resource group, click the [Grant access](#) button.

**Grant access**

[Azure SQL databases and elastic pools with Microsoft Entra authentication](#)

[Azure SQL managed instances with Microsoft Entra authentication](#)

[Azure SQL databases and elastic pools with SQL authentication](#)

[Azure SQL managed instances with SQL authentication](#)

## Add SQL target

Use the dropdowns to select the resource you want to monitor. [Learn more](#) about the types of resources that can be added as targets.

Resource type \* ⓘ SQL database ▼

Subscription \* ⓘ watcher-example ▼

Server \* ⓘ example-database ▼

Connection database \* ⓘ

Read intent ⓘ ☐

To start monitoring, grant the watcher specific access to this target. [Learn more](#)

By default, the managed identity of the watcher is used to authenticate to each target. If you prefer to use SQL authentication, or if Microsoft Entra authentication is not supported or is not enabled, check the box below.

Use SQL authentication ☐

**Add**

# Grant Access for Database Watcher

The image illustrates the steps to grant access for Database Watcher to an SQL database. It is divided into three main sections:

### Grant access to example-watcher-1

1 CREATE LOGIN [example-watcher-1] FROM EXTERNAL PROVIDER;  
2  
3 ALTER SERVER ROLE ##MS\_ServerPerformanceStateReader## ADD MEMBER [example-watcher-1];  
4 ALTER SERVER ROLE ##MS\_DefinitionReader## ADD MEMBER [example-watcher-1];  
5 ALTER SERVER ROLE ##MS\_DatabaseConnector## ADD MEMBER [example-watcher-1];  
6

**COPY**

**Grant access**

- Azure SQL databases and elastic pools with Microsoft Defender for SQL
- Azure SQL managed instances with Microsoft Defender for SQL
- Azure SQL databases and elastic pools with Microsoft Defender for SQL
- Azure SQL managed instances with SQL Server

**SQLQuery1.sql - wa...microsoft.com (61)\*\* - Microsoft SQL Server Management Studio**

File Edit View Query Project Tools Window Help

master Execute

```
CREATE LOGIN [example-watcher-1] FROM EXTERNAL PROVIDER;  
  
ALTER SERVER ROLE ##MS_ServerPerformanceStateReader## ADD MEMBER [example-watcher-1];  
ALTER SERVER ROLE ##MS_DefinitionReader## ADD MEMBER [example-watcher-1];  
ALTER SERVER ROLE ##MS_DatabaseConnector## ADD MEMBER [example-watcher-1];
```

100 %

Messages

Commands completed successfully.

Completion time: 2024-01-27T19:25:30.1937341-05:00

# Create Managed Private Endpoint

Home > example-watcher-1

example-watcher-1 | Managed private endpoints

Database watcher

Search

+ Add

Delete

Refresh

Overview

Activity log

Access control (IAM)

Resource visualizer

Favorites

Monitoring

Configuration

Identity

Managed private endpoints

SQL targets

Data store

Automation

Name

Re

No managed private endpoints found.

Add managed private endpoint

A managed private endpoint request must be approved by the resource owner.

A managed private endpoint enables this watcher to connect to an Azure resource by using private connectivity. [Learn more](#)

Name \*

watcher-example-private-endpoint

Subscription

Resource type \*

microsoft.sql/servers

Resource \*

watcher-example

Target sub-resource \*

sqlServer

Description ⓘ

This private endpoint is for example-watcher-1

Description must be less than 140 characters in length.

Create

# Approve Managed Private Endpoint

Home > Private Link Center

Private Link Center | Pending connections

Search

Refresh

Approve

Reject

Remove

Overview

Pending connections

Private endpoints

Private link services

Azure Arc private link scopes

Azure Monitor private link scopes

Resources

Filter by name...

Subscripti... ==

Name

↑↓

Resource

↑↓

Private endpoint

↑↓

Description

↑↓

watcher-example-pe-0ddacaa0-d4...


SQL

watcher-example

watcher-example-pe

# Start Database Watcher

Home >

 **example-watcher-1** ☆ ...

Database watcher

Search

Start Stop Delete Refresh

Overview

Activity log

Access control (IAM)

Monitoring

Dashboards

Essentials


Resource group : [database-watcher-quickstart](#)

Location : East US

Subscriptions

Status : Stopped

Home > example-watcher-1

 **example-watcher-1** | Dashboards ☆ ...

Database watcher

Search

Auto refresh: Off

Overview

Activity log

Access control (IAM)

Resource visualizer

Favorites

Monitoring

Dashboards

Configuration

Automation

Data store

Azure SQL databases

1

Azure SQL elastic pools

0

Azure SQL managed instances

0

Azure SQL databases

Search

Logical server	Database	Elastic pool	Uptime	Service tier	Compute size	Logical CPUs	Replica type	Age of sample
watcher-example (1)								
watcher-example	example-database	(None)	0.00:49:00	Hyperscale	HS_Gen5_2	2	Primary	0.00:00:50

View [ingestion statistics](#) to see data ingestion latency for each dataset

Learn more

# Demonstration

## Monitor Performance with Database Watcher

- Azure SQL Database monitoring with Database Watcher



Questions?



# Module Summary

