Windows PowerShell Get-Help on Cmdlet 'Get-SqlAssessmentItem'

PS:\>Get-HELP Get-SqlAssessmentItem -Full

NAME

Get-SqlAssessmentItem

SYNOPSIS

Gets SQL Assessment best practice checks available for a chosen SQL Server object.

SYNTAX

Get-SqlAssessmentItem [[-InputObject] <PSObject>] [-Check <String[]>] [-Configuration <PSObject>] [-FlattenOutput] [-MinSeverity {Information | Low | Medium | High}]

[-ProgressAction <ActionPreference>] [<CommonParameters>]

DESCRIPTION

The Get-SqlAssessmentItem cmdlet finds all available best practice checks for each input object. For more information, see the SQL Assessment API overview

(/sql/tools/sql-assessment-api/sql-assessment-api-overview).

This cmdlet accepts the following input types:

- Microsoft.SqlServer.Management.Smo.Server

- Microsoft.SqlServer.Management.Smo.Database

- Microsoft.SqlServer.Management.Smo.AvailabilityGroup

- Microsoft.SqlServer.Management.Smo.FileGroup

- Microsoft.SqlServer.Management.Smo.RegisteredServers.RegisteredServer

- String containing path to any object of the above types

- Collection of objects

You can get input objects with SqlServer cmdlets like Get-SqlInstance and Get-SqlDatabase or basic PowerShell cmdlets like Get-Item and Get-ChildItem. Also, the cmdlet supports the SQL Server PowerShell provider, so it can obtain an object from its path. The path can be passed explicitly, otherwise the current path will be used.

Availability of a check for a chosen object varies on the SQL Server version, platform, and object type. Also, there are checks that target specific databases like `tempdb` or `master`. You can additionally filter checks by tags, names, and severity with the parameters -MinSeverity and -Check.

With Get-SqlAssessmentItem cmdlet, you can get a list of checks applicable to the given SQL Server object. Also, you can use this cmdlet's output as -Check argument for Invoke-SqlAssessment cmdlet.

Custom configurations can be applied with the -Configuration parameter. Customization examples are available on Github (https://go.microsoft.com/fwlink/?linkid=2099023).

SQL Server on Azure VM support

With SQL Assessment cmdlets, you can assess an instance of SQL Server on Azure VM not only as on-prem SQL Server, but also with rules that are specific to SQL Server

on Azure VM (ones that use information about the virtual machine configuration). For example, the AzSqlVmSize rule checks that the VM that hosts an instance of SQL

Server on Azure VM is of recommended size.

To use such rules, connect to Azure with Azure PowerShell Module (/powershell/azure/get-started-azureps) and make sure that the

[Az.ResourceGraph](https://www.powershellgallery.com/packages/Az.ResourceGraph)module is installed. Sign in with Azure PowerShell

(/powershell/azure/authenticate-azureps)before invoking SQL Assessment against a SQL Server on Azure VM instance. Example 13 shows the interactive sign in process

and subscription selection.

NOTE. It is possible to use Azure account connection persisted between PowerShell sessions, i.e. invoke Connect-AzAccount in one session and omit this command later.

However, the current version of SQL Assessment cmdlets needs the Az.ResourceGraph module to be imported explicitly in this case: Import-Module Az.ResourceGraph

PARAMETERS

-Check <String[]>

One or more checks, check IDs, or tags.

For every check object, Get-SqlAssessmentItem returns that check if it supports the input object.

For every check ID, Get-SqlAssessmentItem returns the corresponding check if it supports the input object.

For tags, Get-SqlAssessmentItem returns checks with any of those tags.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-Configuration <PSObject>

Specifies paths to files containing custom configuration. Customization files will be applied to default configuration in specified order. The scope is limited

to this cmdlet invocation only.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-FlattenOutput [<SwitchParameter>]

Indicates that this cmdlet produces simple objects of type Microsoft.SqlServer.Management.Assessment.Cmdlets.AssessmentNoteFlat instead of

Microsoft.SqlServer.Management.Assessment.Cmdlets.AssessmentNote .

Required? false

Position? named

Default value False

Accept pipeline input? False

Accept wildcard characters? false

-InputObject <PSObject>

Specifies a SQL Server object or a path to such an object. The cmdlet returns appropriate checks for this object. When this parameter is omitted, current

location is used as input object. If current location is not a supported SQL Server object, the cmdlet signals an error.

Required? false

Position? 10

Default value None

Accept pipeline input? True (ByValue)

Accept wildcard characters? false

-MinSeverity <SeverityLevel>

Specifies minimum severity level for checks to be found. For example, checks of Medium, Low, or Information levels will not be returned when -MinSeverity High.

Required? false

Position? named

Default value Information

Accept pipeline input? False

Accept wildcard characters? false

-ProgressAction <ActionPreference>

Determines how PowerShell responds to progress updates generated by a script, cmdlet, or provider, such as the progress bars generated by the Write-Progress

cmdlet. The Write-Progress cmdlet creates progress bars that show a command's status.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug,

ErrorAction, ErrorVariable, WarningAction, WarningVariable,

OutBuffer, PipelineVariable, and OutVariable. For more information, see

about_CommonParameters (https:/go.microsoft.com/fwlink/?LinkID=113216).

INPUTS

System.String[]

Microsoft.SqlServer.Management.Smo.SqlSmoObject[]

OUTPUTS

Microsoft.SqlServer.Management.Assessment.ICheck

NOTES

------- Example 1: Get checks for local default instance -------

PS:> Get-SqlInstance -ServerInstance 'localhost' | Get-SqlAssessmentItem

Target: [LOCAL]

ID	ON	Name	Origin
TF1204	True	TF 1204 returns deadlock information	Microsoft Ruleset 0.1.202
BlackboxTrace	True	Blackbox trace is configured and running	Microsoft Ruleset 0.1.202
HintsStatistics	True	Hints are being used	Microsoft Ruleset 0.1.202
PlansUseRatio	True	Amount of single use plans in cache i...	Microsoft Ruleset 0.1.202
TempDBFilesAutoGrowth	True	Some TempDB data files have different...	Microsoft Ruleset 0.1.202
CpuUtil90	True	CPU usage over 90%	Microsoft Ruleset 0.1.202

...

This example gets all checks available for the default instance of SQL Server running on the current machine.

---------- Example 2: Get checks with Get-Item cmdlet ----------

PS:> Get-Item SQLSERVER:\SQL\localhost\default | Get-SqlAssessmentItem

Target: [LOCAL]

ID	ON	Name	Origin
TF1204	True	TF 1204 returns deadlock information	Microsoft Ruleset 0.1.202
BlackboxTrace	True	Blackbox trace is configured and running	Microsoft Ruleset 0.1.202
HintsStatistics	True	Hints are being used	Microsoft Ruleset 0.1.202
PlansUseRatio	True	Amount of single use plans in cache i...	Microsoft Ruleset 0.1.202
TempDBFilesAutoGrowth	True	Some TempDB data files have different...	Microsoft Ruleset 0.1.202
CpuUtil90	True	CPU usage over 90%	Microsoft Ruleset 0.1.202

...

This example gets all checks available for the default instance of SQL Server running on the current machine.

------- Example 3: Get checks with path to target object -------

PS:> Get-SqlAssessmentItem SQLSERVER:\SQL\localhost\default

Target: [LOCAL]

ID	ON	Name	Origin
TF1204	True	TF 1204 returns deadlock information	Microsoft Ruleset 0.1.202
BlackboxTrace	True	Blackbox trace is configured and running	Microsoft Ruleset 0.1.202
HintsStatistics	True	Hints are being used	Microsoft Ruleset 0.1.202
PlansUseRatio	True	Amount of single use plans in cache i...	Microsoft Ruleset 0.1.202
TempDBFilesAutoGrowth	True	Some TempDB data files have different...	Microsoft Ruleset 0.1.202
CpuUtil90	True	CPU usage over 90%	Microsoft Ruleset 0.1.202

...

This example gets all checks available for the default instance of SQL Server running on the current machine.

--- Example 4: Get checks with applied custom configuration ---

```
PS:> Get-SqlDatabase master -ServerInstance . |
    Get-SqlAssessmentItem -Configuration C:\rulesetA.json, D:\rulesetB.json

  Target: [LOCAL]

ID                       ON      Name                            Origin
--                       --      ----                            ------
TF1204                   False   TF 1204 returns deadlock information     Microsoft Ruleset 0.1.202
BlackboxTrace            True    Blackbox trace is configured and running Microsoft Ruleset 0.1.202
HintsStatistics          True    Hints are being used                     Microsoft Ruleset 0.1.202
PlansUseRatio            True    Amount of single use plans in cache i... Microsoft Ruleset 0.1.202
TempDBFilesAutoGrowth    False   Some TempDB data files have different... Microsoft Ruleset 0.1.202
CpuUtil90                True    CPU usage over 90%                       Microsoft Ruleset 0.1.202
SomeCustomCheck          True    Some custom check                        Ruleset A 1.0
AnotherCustomCheck       True    Another custom check                     Ruleset B 1.0
...
```

This example gets all available checks with applied custom configuration obtained from specified JSON files. Visit SQL Assessment samples folder (https://go.microsoft.com/fwlink/?linkid=2099023)on Github to find out how to make customization.

----- Example 5: Get checks for all instances on localhost -----

```
PS:> Get-SqlInstance -ServerInstance localhost | Get-SqlAssessmentItem

  Target: [LOCAL]

ID                       ON      Name                            Origin
--                       --      ----                            ------
TF1204                   True    TF 1204 returns deadlock information     Microsoft Ruleset 0.1.202
```

BlackboxTrace True Blackbox trace is configured and running Microsoft Ruleset 0.1.202

CpuUtil90 True CPU usage over 90% Microsoft Ruleset 0.1.202

Target: [LOCAL\INSTANCE1]

ID ON Name Origin

-- -- ---- ------

HintsStatistics True Hints are being used Microsoft Ruleset 0.1.202

PlansUseRatio True Amount of single use plans in cache i... Microsoft Ruleset 0.1.202

TempDBFilesAutoGrowth True Some TempDB data files have different... Microsoft Ruleset 0.1.202

CpuUtil90 True CPU usage over 90% Microsoft Ruleset 0.1.202

...

This example shows Get-SqlAssessmentItem cmdlet accepting a set of SQL Server instances via pipeline.

Example 6: Get checks for all instances with names ending with numbers

PS:> Get-SqlInstance -ServerInstance localhost | Where { $_.Name -Match '.*\d+' } | Get-SqlAssessmentItem

Target: [LOCAL\INSTANCE1]

ID ON Name Origin

-- -- ---- ------

HintsStatistics True Hints are being used Microsoft Ruleset 0.1.202

PlansUseRatio True Amount of single use plans in cache i... Microsoft Ruleset 0.1.202

TempDBFilesAutoGrowth True Some TempDB data files have different... Microsoft Ruleset 0.1.202

CpuUtil90 True CPU usage over 90% Microsoft Ruleset 0.1.202

...

This example shows Get-SqlAssessmentItem cmdlet accepting a set of SQL Server instances via pipeline. Only instances having the name ending with digits are processed.

--------- Example 7: Get checks for a database by path ---------

PS:> Get-SqlAssessmentItem SQLSERVER:\SQL\localhost\default\Databases\master

TargetObject: [master]

ID	ON	Name	Origin
AutoCreateStats	True	Auto-Create Statistics should be on	Microsoft Ruleset 0.1.202
HintsUsageInModules	False	Hints usage in modules	Microsoft Ruleset 0.1.202
FullBackup	True	Full backup is missed or outdated	Microsoft Ruleset 0.1.202
DuplicateIndexes	True	Duplicate Indexes	Microsoft Ruleset 0.1.202
RedundantIndexes	True	Redundant Indexes	Microsoft Ruleset 0.1.202

...

This example shows Get-SqlAssessmentItem cmdlet accepting a path to a SQL Server database.

------ Example 8: Get high severity checks for a database ------

```
PS:> cd SQLSERVER:\SQL\localhost\default\Databases\master
PS:> Get-SqlAssessmentItem -MinSeverity High
```

This example shows Get-SqlAssessmentItem returning available checks with high severity for the master database. It accepts the current PowerShell provider location

as the target.

------ Example 9: Get high severity checks for a database ------

```
PS:> $db = Get-SqlDatabase master -ServerInstance localhost
PS:> Get-SqlAssessmentItem $db -MinSeverity High
```

This example shows Get-SqlAssessmentItem returning available checks with high severity for the master database.

---------- Example 10: Get checks by tag ----------

```
PS:> Get-SqlDatabase -ServerInstance . | Get-SqlAssessmentItem -Check Backup
```

TargetObject: [master]

ID	ON	Name	Origin
FullBackup	True	Full backup is missed or outdated	Microsoft Ruleset 0.1.202

TargetObject: [msdb]

ID	ON	Name	Origin
FullBackup	True	Full backup is missed or outdated	Microsoft Ruleset 0.1.202

This example shows Get-SqlAssessmentItem cmdlet returning all backup-related checks for all databases on default local SQL Server instance.

-------- Example 11: Run interactively selected checks --------

```
PS:> $serverInstance = Get-SqlInstance -ServerInstance '(local)'
PS:> $checks = Get-SqlAssessmentItem $serverInstance | Select Id, Description | Out-GridView -PassThru
PS:> Invoke-SqlAssessment $serverInstance -Check $checks
```

TargetPath : Server[@Name='LOCAL']

Sev.	Message	Check ID	Origin
Info	Enable trace flag 834 to use large-page allocations to improve analytical and data warehousing workloads.	TF834	Microsoft Ruleset 0.1.202
Low	Detected deprecated or discontinued feature uses: String literals as column aliases, syscolumns, sysusers, SET FMTONLY ON, XP_API, Table hint without WITH, More than two-part column name. We recommend to replace them with features actual for SQL Server version 14.0.1000.	DeprecatedFeatures	Microsoft Ruleset 0.1.202

The second line of this example shows obtaining checks for a $serverInstance, and selecting some of them interactively. Selected items are stored in an array

variable, which then can be used as input for Invoke-SqlAssessment cmdlet. In this case, only picked checks will run during the assessment process.

---------- Example 12: Specify credentials explicitly ----------

```
PS> $cred = Get-Credential

PowerShell credential request
Enter your credentials.
User: Administrator
Password for user Administrator: ********

PS> $db = Get-SqlDatabase master -ServerInstance 10.0.3.118 -Credential $cred
PS> Get-SqlAssessmentItem $db

  TargetObject: [master]

ID                      ON    Name                          Origin
--                      --    ----                          ------
AutoCreateStats          True  Auto-Create Statistics should be on  Microsoft Ruleset 0.1.202
FullBackup               True  Full backup is missed or outdated    Microsoft Ruleset 0.1.202
DuplicateIndexes         True  Duplicate Indexes                    Microsoft Ruleset 0.1.202
RedundantIndexes         True  Redundant Indexes                    Microsoft Ruleset 0.1.202
...
```

This example shows how to get the SQL Assessment check list with explicitly specified credentials.

Example 13: Get the SQL Assessment rule list for the SQL Server on Azure VM instance

```
PS> Connect-AzAccount
PS> Set-Subscription My-Pay-As-You-Go
PS> $cred = Get-Credential
```

PowerShell credential request

Enter your credentials.

User: Administrator

Password for user Administrator: ********

PS> $inst = Get-SqlInstance -ServerInstance 10.0.3.118 -Credential $cred

PS> Get-SqlAssessmentItem $inst

TargetObject: [ContosoAzureSql]

ID	ON	Name	Origin
HintsStatistics	True	Hints are being used	Microsoft Ruleset 0.1.202
PlansUseRatio	True	Amount of single use plans in cache i...	Microsoft Ruleset 0.1.202
TempDBFilesAutoGrowth	True	Some TempDB data files have different...	Microsoft Ruleset 0.1.202
AzSqlVmSize	True	VM size is not memory-optimized	Microsoft Ruleset 0.1.202

...

This example shows how to get a list of rules that are applicable to a particular SQL Server on Azure VM instance.

An active Azure subscription connection enables Azure-related checks (AzSqlVmSize in this example). The first line connects to an Azure account to get data from Azure

Resource Graph. The second line is optional.

To run these checks, SQL Assessment requires the Az.ResourceGraph module.

RELATED LINKS

Online Version: https://learn.microsoft.com/powershell/module/sqlserver/get-sqlassessmentitem

SQL Assessment API Online Documentation page

SQL Assessment Samples https://go.microsoft.com/fwlink/?linkid=2099023