Windows PowerShell Get-Help on Cmdlet 'Invoke-SqlAssessment'

PS:\>Get-HELP Invoke-SqlAssessment -Full

NAME

Invoke-SqlAssessment

SYNOPSIS

Runs SQL Assessment best practice checks for a chosen SQL Server object and returns their results.

SYNTAX

Invoke-SqlAssessment [[-InputObject] <PSObject>] [-Check <Object[]>] [-Configuration <PSObject>] [-FlattenOutput] [-MinSeverity {Information | Low | Medium | High}]

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

DESCRIPTION

The Invoke-SqlAssessment cmdlet runs an assessment for each input object and returns a list of best practice recommendations that should be applied to the specified

objects. It's up to you to follow the given recommendations or not. 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.

You can get a list of checks applicable to the given SQL Server object with Get-SqlAssessmentItem cmdlet.

The cmdlet runs only checks that are applicable to an input object. For example, database checks will not be run for a SQL Server instance or an availability group,

even when specified in -Check list.

Custom configurations can be applied with the -Configuration parameter. Customization examples are available on Github

(https://go.microsoft.com/fwlink/?linkid=2099023).

NOTE. In the first public preview Invoke-SqlAssessment returned AssessmentNote objects with properties CheckId and CheckName containing Check's ID and DisplayName

respectively. In the second public preview the two properties were replaced with a single Check property providing much more data. Assuming $note was an object

returned by Invoke-SqlAssessment , you can access check's ID as $note.Check.Id instead of $note.CheckId, or check's description as $note.Check.Description. You can

use -FlattenOutput parameter to get results in the previous format with CheckId and CheckName. This parameter will also help to retain compatibility to some cmdlets

like Write-SqlTableData . See examples 12-14 for more details. Invoke-SqlAssessment cmdlet's output is a list of violated best practices for every given SQL Server

object. Use Description property to learn about the best practice and Message property to find out how it can be solved. Also, every check result contains a link to

online documentation, which will help you figure out the issue better.

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 16 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 <Object[]>

One or more checks, check IDs, or tags.

For every check object, Invoke-SqlAssessment runs that check if it supports the input object.

For every check ID, Invoke-SqlAssessment runs the corresponding check if it supports the input object.

For tags, Invoke-SqlAssessment runs 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 .

Regular AssessmentNote returned from Invoke-SqlAssessment contains references to other useful complex objects like Check (see example 12). With the Check property, you can get the check's description or reuse the check (see example 13). But some cmdlets do not support complex properties. For example, Write-SqlTableData will raise an error while trying to write AssessmentNote to a database. To avoid this you can use -FlattenOutput parameter to replace the Check property with two simple strings: CheckId and CheckName (see example 14).

Required? false

Position? named

Default value False

Accept pipeline input? False

Accept wildcard characters? false

-InputObject <PSObject>

Specifies a SQL Server object or the path to such an object. The cmdlet runs assessment 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 Low, Medium or Information levels will not be returned when -MinSeverity High.

Required? false

Position? named

Default value None

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.Assessment.Cmdlets.AssessmentNote

NOTES

--- Example 1: Invoke assessment for local default instance ---

PS:> Get-SqlInstance -ServerInstance localhost | Invoke-SqlAssessment

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
Medi	Amount of single use plans in cache is high (100%). Consider enabling the Optimize for ad hoc workloads setting on heavy OLTP ad-hoc workloads to conserve resources.	PlansUseRatio	Microsoft Ruleset 0.1.202

...

This example shows how to get all best practice recommendations for the default instance of SQL Server running on the current machine.

----- Example 2: Invoke assessment with PSProvider cmdlet -----

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

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
Medi	Amount of single use plans in cache is high (100%). Consider enabling the Optimize for ad hoc workloads setting on heavy OLTP ad-hoc workloads to conserve resources.	PlansUseRatio	Microsoft Ruleset 0.1.202

...

This example shows how to get all best practice recommendations for the default instance of SQL Server.

------ Example 3: Invoke assessment with PS Provider path ------

PS:> Invoke-SqlAssessment SQLSERVER:\SQL\localhost\default

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,	DeprecatedFeatures	Microsoft Ruleset 0.1.202

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.

Medi Amount of single use plans in cache is high (100%). Consider PlansUseRatio Microsoft Ruleset 0.1.202
enabling the Optimize for ad hoc workloads setting on heavy OLTP
ad-hoc workloads to conserve resources.

...

This example shows how to get all best practice recommendations for the default instance of SQL Server.

---- Example 4: Invoke assessment with custom configuration ----

PS:> Get-SqlInstance -ServerInstance '(local)' | Invoke-SqlAssessment -Configuration C:\profileA.json, C:\profileB.json

TargetPath : Server[@Name='LOCAL']

Sev. Message Check ID Origin

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

Low Detected deprecated or discontinued feature uses: String literals DeprecatedFeatures Microsoft Ruleset 0.1.202
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.

Medi A custom rule violation detected. CustomRuleA Profile A 1.0

...

This example shows how to apply custom configuration to get a modified set of best practice recommendations. Custom configurations are described in JSON files.

Custom rulesets profileA.json and profileB.json disabled some checks and introduced some new ones. One of the new checks from profileA.json detected a problem with
the current configuration of the SQL Server instance. Visit SQL Assessment samples folder (https://go.microsoft.com/fwlink/?linkid=2099023)on Github to find out how
to make customization.

-------- Example 5: Invoke assessment for all instances --------

```
PS:> dir SQLSERVER:\SQL\localhost | Invoke-SqlAssessment

    TargetPath : Server[@Name='LOCAL']

Sev. Message                                                    Check ID            Origin
---- -------                                                    --------            ------
Info Enable trace flag 834 to use large-page allocations to improve     TF834               Microsoft Ruleset 0.1.202
     analytical and data warehousing workloads.
Low  Detected deprecated or discontinued feature uses: String literals  DeprecatedFeatures    Microsoft Ruleset 0.1.202
     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.
Medi Amount of single use plans in cache is high (100%). Consider       PlansUseRatio       Microsoft Ruleset 0.1.202
     enabling the Optimize for ad hoc workloads setting on heavy OLTP
     ad-hoc workloads to conserve resources.

    TargetPath : Server[@Name='LOCAL\INSTANCE1']

Sev. Message                                                    Check ID            Origin
---- -------                                                    --------            ------
Medi Product version 14.0.1000 is not the latest available. Keep your   LatestCU            Microsoft Ruleset 0.1.202
     your SQL Server up to date and install Service Packs and
     Cumulative Updates as they are released.
...
```

This example shows Invoke-SqlAssessment cmdlet accepting a set of SQL Server instances via pipeline.

--- Example 6: Run assessment for a filtered set ot tragets ---

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

    TargetPath : Server[@Name='LOCAL\INSTANCE1']
```

Sev. Message Check ID Origin

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

Medi Product version 14.0.1000 is not the latest available. Keep your LatestCU Microsoft Ruleset 0.1.202

your SQL Server up to date and install Service Packs and

Cumulative Updates as they are released.

...

This example shows Invoke-SqlAssessment cmdlet accepting a set of SQL Server instances via pipeline. The set is filtered with the standard PowerShell Where-Object

cmdlet.

----- Example 7: Invoke assessment for a database by name -----

PS:> Get-SqlDatabase master -ServerInstance localhost | Invoke-SqlAssessment -Verbose

VERBOSE: Base ruleset version: 0.2.

VERBOSE: No recommendations for [master].

This example shows Invoke-SqlAssessment cmdlet accepting name of a database. In this case no issue was found.

----- Example 8: Invoke assessment for a database by path -----

PS:> Invoke-SqlAssessment SQLSERVER:\SQL\localhost\default\Databases\master -Verbose

VERBOSE: Base ruleset version: 0.2.

VERBOSE: No recommendations for [master].

This example shows Invoke-SqlAssessment cmdlet accepting the path to a SQL Server database.

--------- Example 9: Detect high issues for a database ---------

PS:> cd SQLSERVER:\SQL\localhost\default\Databases\master

PS:> Invoke-SqlAssessment -MinSeverity High

VERBOSE: Base ruleset version: 0.2.

VERBOSE: No recommendations for [master].

This example shows Invoke-SqlAssessment cmdlet assessing the current location. Only high issues are reported.

---------- Example 10: Run checks selected by tag ----------

PS:> Get-SqlInstance -ServerInstance . | Invoke-SqlAssessment -Check Backup -Verbose

VERBOSE: Base ruleset version: 0.2.

VERBOSE: No recommendations for [LOCAL].

This example shows Invoke-SqlAssessment cmdlet running all backup-related checks for every SQL Server instance on the local server.

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

PS:> $serverInstance = Get-SqlInstance -ServerInstance '(local)'

PS:> $checks = Get-SqlAssessmentItem $serverInstance | Select Name, 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 is used as input for Invoke-SqlAssessment cmdlet. Only picked checks run during the assessment process.

-------- Example 12: Effect of -FlattenOutput parameter --------

PS> $inst = Get-SqlInstance -ServerInstance .

```
PS> Invoke-SqlAssessment $inst -FlattenOutput | Select -First 1 | Get-Member


   TypeName: Microsoft.SqlServer.Management.Assessment.Cmdlets.AssessmentNoteFlat

Name           MemberType Definition
----           ---------- ----------
Equals         Method     bool Equals(System.Object obj)
GetHashCode    Method     int GetHashCode()
GetType        Method     type GetType()
ToString       Method     string ToString()
CheckId        Property   string CheckId {get;}
CheckName      Property   string CheckName {get;}
HelpLink       Property   string HelpLink {get;}
Message        Property   string Message {get;}
RulesetName    Property   string RulesetName {get;}
RulesetVersion Property   string RulesetVersion {get;}
Severity       Property   string Severity {get;}
TargetPath     Property   string TargetPath {get;}
TargetType     Property   string TargetType {get;}


PS> Invoke-SqlAssessment $inst | Select -First 1 | Get-Member


   TypeName: Microsoft.SqlServer.Management.Assessment.Cmdlets.AssessmentNote

Name        MemberType Definition
----        ---------- ----------
Equals      Method     bool Equals(System.Object obj)
GetHashCode Method     int GetHashCode()
GetType     Method     type GetType()
ToString    Method     string ToString()
```

```
Check       Property   Microsoft.SqlServer.Management.Assessment.Checks.ICheck Check {get;}
HelpLink    Property   string HelpLink {get;}
Message     Property   string Message {get;}
Severity    Property   Microsoft.SqlServer.Management.Assessment.SeverityLevel Severity {get;}
TargetPath  Property   string TargetPath {get;}
TargetType  Property   string TargetType {get;}

PS> (Invoke-SqlAssessment $inst | Select -First 1).Check | Get-Member


   TypeName: Microsoft.SqlServer.Management.Assessment.Checks.Check

Name          MemberType Definition
----          ---------- ----------
Equals        Method     bool Equals(System.Object obj)

GetHashCode   Method     int GetHashCode()

GetType       Method     type GetType()

ToString      Method     string ToString()

Condition     Property   Microsoft.SqlServer.Management.Assessment....
Description   Property   string Description {get;set;}

DisplayName   Property   string DisplayName {get;set;}

Enabled       Property   bool Enabled {get;set;}
```

HelpLink Property string HelpLink {get;set;}

Id Property string Id {get;set;}

Level Property Microsoft.SqlServer.Management.Assessment....

Message Property string Message {get;set;}

OriginName Property string OriginName {get;set;}

OriginVersion Property version OriginVersion {get;set;}

Parameters Property System.Collections.Generic.IDictionary[str...

Probes Property System.Collections.Generic.List[Microsoft....

Tags Property System.Collections.Generic.HashSet[string]...

Target Property Microsoft.SqlServer.Management.Assessment....

This example shows the difference between objects returned with or without -FlattenOutput parameter. The parameter replaces huge complex Check object with two string

properties CheckId and CheckName. This is useful for serialization purposes (see the next example).

The first command shows simple object with all properties of type string.

The second command shows another object with two non-string properties: Check and Severity.

The third command shows the rich set of data accessible with the Check property.

---------- Example 13: Run failed checks again ----------

PS> $inst = Get-SqlInstance -ServerInstance .

PS> $results = Invoke-SqlAssessment $inst

```
PS> $results

   TargetPath : Server[@Name='LOCAL']

Sev. Message                                                    Check ID            Origin
---- -------                                                    --------            ------
Info Enable trace flag 834 to use large-page allocations to improve     TF834               Microsoft Ruleset 0.1.202
     analytical and data warehousing workloads.
Low  Detected deprecated or discontinued feature uses: String literals  DeprecatedFeatures   Microsoft Ruleset 0.1.202
     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.
Medi Amount of single use plans in cache is high (100%). Consider       PlansUseRatio       Microsoft Ruleset 0.1.202
     enabling the Optimize for ad hoc workloads setting on heavy OLTP
     ad-hoc workloads to conserve resources.

PS> $results[1].Check.Description
 This check detects deprecated or discontinued features used on target SQL Server instance. Deprecated features may
be
 removed in a future release of SQL Server. Discontinued features have been removed from specific versions of SQL
Server.

PS> Invoke-SqlAssessment $inst -Check $results[1].Check

   TargetPath : Server[@Name='LOCAL']

Sev. Message                                                    Check ID            Origin
---- -------                                                    --------            ------

Low  Detected deprecated or discontinued feature uses: String literals  DeprecatedFeatures   Microsoft Ruleset 0.1.202
     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.

PS> Invoke-SqlAssessment $inst -Check ($results).Check

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
Medi	Amount of single use plans in cache is high (100%). Consider enabling the Optimize for ad hoc workloads setting on heavy OLTP ad-hoc workloads to conserve resources.	PlansUseRatio	Microsoft Ruleset 0.1.202

This example shows how a Check returned with assessment results can be reused. You can rerun the checks which produced AssessmentNotes before.

------- Example 14: Store assessment results to a table -------

PS:> $serverInstance = Get-SqlInstance -ServerInstance '(local)'
PS:> Invoke-SqlAssessment $serverInstance -FlattenOutput |
Write-SqlTableData -ServerInstance localhost -DatabaseName SqlAssessment -SchemaName Assessment -TableName Results -Force

The second line of this example shows using -FlattenOutput parameter with Write-SqlTableData cmdlet to store assessment results to a SQL database.

---------- Example 15: Specify credentials explicitly ----------

```
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> Invoke-SqlAssessment $inst

   TargetPath : Server[@Name='ContosSQL']

Sev. Message                                          Check ID            Origin
---- -------                                          --------            ------
Medi Amount of single use plans in cache is high (100%). Consider       PlansUseRatio       Microsoft Ruleset 0.1.202
     enabling the Optimize for ad hoc workloads setting on heavy OLTP
     ad-hoc workloads to conserve resources
```

This example shows how to invoke SQL Assessment with explicitly specified credentials.

Example 16: Invoke assessment for a 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> Invoke-SqlAssessment $inst
```

TargetPath : Server[@Name='ContosoAzureSQL']

Sev.	Message	Check ID	Origin
Medi	Amount of single use plans in cache is high (100%). Consider enabling the Optimize for ad hoc workloads setting on heavy OLTP ad-hoc workloads to conserve resources	PlansUseRatio	Microsoft Ruleset 0.1.202
Info	Use memory optimized virtual machine sizes for the best performance of SQL Server workloads	AzSqlVmSize	Microsoft Ruleset 0.1.202

This example shows how to invoke assessment for an SQL Server on Azure VM instance.

An active Azure subscription connection enables SQL Server on Azure VM 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 Az.ResourceGraph module.

RELATED LINKS

Online Version: https://learn.microsoft.com/powershell/module/sqlserver/invoke-sqlassessment

SQL Assessment API Online Documentation page

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