Windows PowerShell Get-Help on Cmdlet 'Add-SqlAvailabilityDatabase'

PS:\>Get-HELP Add-SqlAvailabilityDatabase -Full

NAME

Add-SqlAvailabilityDatabase

SYNOPSIS

Adds primary databases to an availability group or joins secondary databases to an availability group.

SYNTAX

Add-SqlAvailabilityDatabase [-InputObject] <AvailabilityGroup[]> [-AccessToken <PSObject>] -Database <String[]> [-Encrypt {Mandatory | Optional | Strict}]

[-HostNameInCertificate <String>] [-ProgressAction <ActionPreference>] [-Script] [-TrustServerCertificate] [-Confirm] [-WhatIf] [<CommonParameters>]

Add-SqlAvailabilityDatabase [[-Path] <String[]>] [-AccessToken <PSObject>] -Database <String[]> [-Encrypt {Mandatory | Optional | Strict}] [-HostNameInCertificate

<String>] [-ProgressAction <ActionPreference>] [-Script] [-TrustServerCertificate] [-Confirm] [-WhatIf] [<CommonParameters>]

DESCRIPTION

The Add-SqlAvailabilityDatabase cmdlet adds primary databases to an availability group or joins secondary databases to an availability group. The InputObject or Path

parameter specifies the availability group. A database can belong to only one availability group.

To add databases to an availability group, run this cmdlet on the server instance that hosts the primary replica. Specify one or more local user databases.

To join a secondary database to the availability group, manually prepare the secondary database on the server instance that hosts the secondary replica. Then run this

cmdlet on the server instance that hosts the secondary replica.

> `Module requirements: version 21+ on PowerShell 5.1; version 22+ on PowerShell 7.x.`

PARAMETERS

-AccessToken <PSObject>

The access token used to authenticate to SQL Server, as an alternative to user/password or Windows Authentication.

This can be used, for example, to connect to `SQL Azure DB` and `SQL Azure Managed Instance` using a `Service Principal` or a `Managed Identity`.

The parameter to use can be either a string representing the token or a `PSAccessToken` object as returned by running `Get-AzAccessToken -ResourceUrl

https://database.windows.net`.

> This parameter is new in v22 of the module.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-Database <String[]>

Specifies an array of user databases. This cmdlet adds or joins the databases that this parameter specifies to the availability group. The databases that you

specify must reside on the local instance of SQL Server.

Required? true

Position? named

Default value None

Accept pipeline input? True (ByPropertyName, ByValue)

Accept wildcard characters? false

-Encrypt <String>

The encryption type to use when connecting to SQL Server.

This value maps to the `Encrypt` property `SqlConnectionEncryptOption` on the SqlConnection object of the Microsoft.Data.SqlClient driver.

In v22 of the module, the default is `Optional` (for compatibility with v21). In v23+ of the module, the default value will be 'Mandatory', which may create a

breaking change for existing scripts.

> This parameter is new in v22 of the module.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-HostNameInCertificate <String>

The host name to be used in validating the SQL Server TLS/SSL certificate. You must pass this parameter if your SQL Server instance is enabled for Force

Encryption and you want to connect to an instance using hostname/shortname. If this parameter is omitted then

passing the Fully Qualified Domain Name (FQDN) to

-ServerInstance is necessary to connect to a SQL Server instance enabled for Force Encryption.

> This parameter is new in v22 of the module.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-InputObject <AvailabilityGroup[]>

Specifies availability group, as an AvailabilityGroup object, to which this cmdlet adds or joins databases.

Required? true

Position? 1

Default value None

Accept pipeline input? True (ByValue)

Accept wildcard characters? false

-Path <String[]>

Specifies the path of an availability group to which this cmdlet adds or joins databases. If you do not specify this parameter, this cmdlet uses current working

location.

Required? false

Position? 1

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

-Script [<SwitchParameter>]

Indicates that this cmdlet returns a Transact-SQL script that performs the task that this cmdlet performs.

Required? false

Position? named

Default value False

Accept pipeline input? False

Accept wildcard characters? false

-TrustServerCertificate [<SwitchParameter>]

Indicates whether the channel will be encrypted while bypassing walking the certificate chain to validate trust.

In v22 of the module, the default is `$true` (for compatibility with v21). In v23+ of the module, the default value will be '$false', which may create a breaking

change for existing scripts.

> This parameter is new in v22 of the module.

Required? false

Position? named

Default value False

Accept pipeline input? False

Accept wildcard characters? false

-Confirm [<SwitchParameter>]

Prompts you for confirmation before running the cmdlet.

Required? false

Position? named

Default value False

Accept pipeline input? False

Accept wildcard characters? false

-WhatIf [<SwitchParameter>]

Shows what would happen if the cmdlet runs. The cmdlet is not run.

Required? false

Position? named

Default value False

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.AvailabilityGroup[]

OUTPUTS

System.Object

NOTES

------ Example 1: Add a database to an availability group ------

PS C:\> Add-SqlAvailabilityDatabase -Path "SQLSERVER:\SQL\PrimaryServer\InstanceName\AvailabilityGroups\MainAG" -Database "Database16"

This command adds the database 'Database16' to the availability group 'MainAG'. Run this command on the primary server instance of the availability group. This

command does not prepare secondary databases for data synchronization.

----- Example 2: Join a database to an availability group -----

PS C:\> Add-SqlAvailabilityDatabase -Path "SQLSERVER:\SQL\SecondaryServer\InstanceName\AvailabilityGroups\MainAG" -Database "Database16"

This command joins a secondary database named 'Database16' to the availability group 'MainAG' on one of the server instances that hosts a secondary replica.

Example 3: Add a database and join a secondary database to an availability group

PS C:\> $DatabaseBackupFile = "\\share\backups\Database16.bak"

PS C:\> $LogBackupFile = "\\share\backups\Database16.trn"

PS C:\> $AGPrimaryPath = "SQLSERVER:\SQL\PrimaryServer\InstanceName\AvailabilityGroups\MainAG"

PS C:\> $MyAGSecondaryPath = "SQLSERVER:\SQL\SecondaryServer\InstanceName\AvailabilityGroups\MainAG"

PS C:\> Backup-SqlDatabase -Database "Database16" -BackupFile $DatabaseBackupFile -ServerInstance "PrimaryServer\InstanceName"

PS C:\> Backup-SqlDatabase -Database "Database16" -BackupFile $LogBackupFile -ServerInstance "PrimaryServer\InstanceName" -BackupAction Log

PS C:\> Restore-SqlDatabase -Database "Database16" -BackupFile $DatabaseBackupFile -ServerInstance "SecondaryServer\InstanceName" -NoRecovery

PS C:\> Restore-SqlDatabase -Database "Database16" -BackupFile $LogBackupFile -ServerInstance "SecondaryServer\InstanceName" -RestoreAction Log -NoRecovery

PS C:\> Add-SqlAvailabilityDatabase -Path $AGPrimaryPath -Database 'Database16'

PS C:\> Add-SqlAvailabilityDatabase -Path $AGSecondaryPath -Database "Database16"

This example prepares a secondary database from a database on the server instance that hosts the primary replica of an availability group. It adds the database to an

availability group as a primary database.Finally, it joins the secondary database to the availability group.

The sixth command backs up the log file for 'Database16' on the primary server to the location in $LogBackupFile.

Example 4: Create a script to add a database to an availability group

PS C:\> Add-SqlAvailabilityDatabase -Path "SQLSERVER:\SQL\PrimaryServer\InstanceName\AvailabilityGroups\MainAG" -Database "Database16" -Script

This command creates a Transact-SQL script that adds the database 'Database16' to the availability group 'MainAG'.

RELATED LINKS

Online Version: https://learn.microsoft.com/powershell/module/sqlserver/add-sqlavailabilitydatabase

Backup-SqlDatabase

Remove-SqlAvailabilityDatabase

Restore-SqlDatabase

Resume-SqlAvailabilityDatabase

Suspend-SqlAvailabilityDatabase