Invoke-DatabaseBuild

Invoke-DatabaseBuild

Builds a database project.

Syntax

```
Invoke-DatabaseBuild [-InputObject] <string> [-TemporaryDatabaseServer <DatabaseServerConnection>] [-
SQLCompareOptions <string>] [-SQLDataCompareOptions <string>] [-TransactionIsolationLevel
<TransactionIsolationLevel>] [-FilterPath <string>] [-QueryBatchTimeout <int>] [-IgnoreParserErrors]
[<CommonParameters>]

Invoke-DatabaseBuild [-InputObject] <string> [-TemporaryDatabase <DatabaseConnection>] [-SQLCompareOptions
<string>] [-SQLDataCompareOptions <string>] [-TransactionIsolationLevel <TransactionIsolationLevel>] [-
FilterPath <string>] [-QueryBatchTimeout <int>] [-IgnoreParserErrors] [<CommonParameters>]
```

Description

The Invoke-DatabaseBuild cmdlet builds a database project by checking the database definition can be deployed to an empty database. It the build is successful, the cmdlet creates an IProject object that represents the built project.

The cmdlet will throw an exception if the project cannot be deployed to an empty database.

By default, the cmdlet creates a temporary copy of the database on LocalDB. Alternatively, you can use the TemporaryDatabaseServer parameter to specify a SQL Server instance for the temporary database. This is useful if your database uses features that aren't supported by LocalDB, such as Full-Text Search.

If you don't want to use LocalDB and don't have permission to create a database on the SQL Server instance, you can use the TemporaryDatabase parameter to specify an existing database.

Parameters

`-InputObject <System.String>`

The path to the database project to validate.

Aliases	None
Required?	true
Position?	0
Default Value	None
Accept Pipeline Input	true (ByValue)
Accept Wildcard Characters	false

`-TemporaryDatabaseServer <RedGate.Versioning.Automation.Compare.SchemaSources.DatabaseServerConnection>`

The connection string for the temporary database server used for validation. For example, 'Data Source=TempServer01'.

By default, LocalDB is used for the temporary database. However there may be some features in your database that aren't supported by LocalDB (for example, Full-Text Search). In this case, or if LocalDB isn't present, use this parameter to specify an alternative SQL Server instance for the temporary database.

Using this option, SQL Change Automation will create a temporary database on the specified SQL Server instance.

You can't use this parameter in addition to the TemporaryDatabase parameter.

Aliases	None
Required?	false
Position?	named
Default Value	None
Accept Pipeline Input	false

Accept Wildcard Characters	false

-TemporaryDatabase <RedGate.Versioning.Automation.Compare.SchemaSources.DatabaseConnection>

The details of the temporary database used for validation. This can be:

- an instance of a Database Connection object produced by the New-DatabaseConnection cmdlet.

- a database connection string. For example, 'Data Source=TempServer01;Initial Catalog=TempDatabase01'.

By default, LocalDB is used for the temporary database. If you don't want to use LocalDB and don't have permission to create a database on the SQL Server instance, use this option to specify an existing database to use for the temporary copy of the database.

If you use this parameter, all existing data on the temporary database will be lost.

You can't use this parameter in addition to the TemporaryDatabaseServer parameter.

Aliases	None
Required?	false
Position?	named
Default Value	None
Accept Pipeline Input	false
Accept Wildcard Characters	false

-SQLCompareOptions <System.String>

Specifies the SQL Compare options to use when creating the script for validation. The default set of options are listed below. To include additional options, specify a comma-delimited list of the option names (eg 'IgnoreComments, ObjectExistenceChecks'). To turn off a default option, precede the option name with a minus sign (eg '-ForceColumnOrder').

This parameter will be ignored if the value specified is $null or empty.

By default, the following Compare options are used:

- ConsiderNextFilegroupInPartitionSchemes

- DecryptPost2KEncryptedObjects

- DoNotOutputCommentHeader

- ForceColumnOrder

- IgnoreCertificatesAndCryptoKeys

- IgnoreDatabaseAndServerName

- IgnoreUsersPermissionsAndRoleMemberships

- IgnoreUserProperties

- IgnoreWhiteSpace

- IgnoreWithElementOrder

- IncludeDependencies

- ThrowOnFileParseFailed

- UseCompatibilityLevel

For more information about SQL Compare options, see http://www.red-gate.com/sca/ps/help/compareoptions.

Aliases	None
Required?	false
Position?	named
Default Value	None
Accept Pipeline Input	false
Accept Wildcard Characters	false

-SQLDataCompareOptions <System.String>

Specifies the SQL Data Compare options to use when creating the script for validation. To include additional options, specify a comma-delimited list of the option names (eg 'DisableAndReenableDDLTriggers, CompressTemporaryFiles').

Aliases	None
Required?	false
Position?	named
Default Value	None
Accept Pipeline Input	false
Accept Wildcard Characters	false

-TransactionIsolationLevel <RedGate.Versioning.Automation.Shared.Domain.TransactionIsolationLevel>

Use this parameter to specify the isolation level for the transactions during the temporary deployment for validation. Permitted values are: Serializable, Snapshot, RepeatableRead, ReadCommitted and ReadUncommitted. The default level is Serializable.

See http://msdn.microsoft.com/en-gb/library/ms173763.aspx for more details on transaction isolation levels.

Possible values: Serializable, Snapshot, RepeatableRead, ReadCommitted, ReadUncommitted

Aliases	None
Required?	false
Position?	named
Default Value	Serializable
Accept Pipeline Input	false
Accept Wildcard Characters	false

-FilterPath <System.String>

The path to a .scpf filter file.

Overrides any Filter.scpf file present in the InputObject schema with an alternative filter file to be used when validating the schema.

This parameter will be ignored if the value specified is either $null or empty.

Aliases	None
Required?	false
Position?	named
Default Value	None
Accept Pipeline Input	false
Accept Wildcard Characters	false

-QueryBatchTimeout <System.Int32>

The execution timeout, in seconds, for each batch of queries in the script for validation. The default value is 30 seconds. A value of zero indicates that no execution timeout will be enforced.

Aliases	None
Required?	false
Position?	named
Default Value	30
Accept Pipeline Input	false
Accept Wildcard Characters	false

-IgnoreParserErrors <System.Management.Automation.SwitchParameter>

Tells the SQL Compare engine to ignore parser errors.

Aliases	None
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, -OutBuffer, and -OutVariable. For more information, see http://technet.microsoft.com/en-us/library/hh847884.aspx.

Inputs

The input type is the type of the objects that you can pipe to the cmdlet.

- **System.String**

  The path to the database project to validate.

Return values

The output type is the type of the objects that the cmdlet emits.

- **RedGate.Versioning.Automation.Compare.Domain.Projects.IProject**

Examples

---------- EXAMPLE 1 ----------

```
$project = "C:\Work\project\project.sqlproj"
$validatedProject = $project | Invoke-DatabaseBuild
```

This example shows how to use the Invoke-DatabaseBuild cmdlet to build a SQL Change Automation project.

A temporary copy of the database is created on LocalDB. If the project content is invalid, this operation will throw an exception. Otherwise, it will output an instance of an IProject object that can be used as the input for other SQL Change Automation cmdlets.

---------- EXAMPLE 2 ----------

```
$project = "C:\Work\scripts"
$validatedProject = $project | Invoke-DatabaseBuild
```

This example shows how to use the Invoke-DatabaseBuild cmdlet to validate the schema and static data of a SQL Source Control project.

A temporary copy of the database is created on LocalDB. If the project content is invalid, this operation will throw an exception. Otherwise, it will output an instance of an IProject object that can be used as the input for other SQL Change Automation cmdlets.

---------- EXAMPLE 3 ----------

```
$project = "C:\Work\project\project.sqlproj"
$validatedProject = $project | Invoke-DatabaseBuild -TemporaryDatabaseServer "Data Source=temp01\sql2014"
```

This example shows how to specify a SQL Server instance for the temporary database, instead of using LocalDB.

This is useful if your database uses features that aren't supported by LocalDB, such as Full-Text Search.

---------- EXAMPLE 4 ----------

```
$project = "C:\Work\project\project.sqlproj"
$temporaryDatabase = New-DatabaseConnection -ServerInstance "temp01\sql2014" -Database "TemporaryDb"
$validatedProject = $project | Invoke-DatabaseBuild -TemporaryDatabase $temporaryDatabase
```

This example shows how to specify an existing SQL Server database to be used when building a project.

This is useful if you don't want to use LocalDB and you don't have permission to create a database on the SQL Server instance.

---------- EXAMPLE 5 ----------

```
$project = "C:\Work\scripts"
$options = "IgnoreComments, ObjectExistenceChecks, -IgnoreUserProperties"
$validatedProject = $project | Invoke-DatabaseBuild -SQLCompareOptions $options
```

This example shows how to specify SQL Compare options to be used when building a SQL Source Control project.

The $options variable is used to specify that the IgnoreComments and ObjectExistenceChecks options should be included in addition to the default set of SQL Compare options used by this cmdlet. The minus sign before IgnoreUserProperties indicates that this default option will be turned off.