Using the sqlCI.exe command line in SQL Automation Pack v1.1 and later

> **Warning**
>
> SQL Automation Pack v1.1.x.x releases contain a rewritten version of SQL CI which is not compatible with 1.0.x.x versions. If you're upgrading from a 1.0.x.x version, you must reconfigure your existing SQL CI build steps after installation. For details of how to upgrade for different CI systems, see Upgrading.

> This page applies to SQL Automation Pack v1.1 and later. If you're using an earlier version, see Using the SqlCI.exe command line in SQL Automation Pack v1.0.4.2 and earlier.

SQL CI supports four steps that include default command line options to perform different continuous integration tasks. The steps are:

1. **Build** - creates and validates the SQL creation script used to generate the NuGet package. See Build step command line options.
2. **Test** - generates test data using SQL Data Generator and runs tSQLt tests against the NuGet package. Results are output in JUnit XML format. See Test step command line options.
3. **Sync** - updates the existing database with the latest version in source control. See Sync step command line options.
4. **Publish** - publishes the NuGet package to a NuGet feed ready for deployment. See Publish step command line options.

> The test, sync and publish steps use the NuGet package created during the build step. These three steps are optional, but if you want to use them you must complete the build step first.

There's also a command to activate your SQL Automation Pack license without having to configure a build, test or sync step. See Activate command line option.

Contents:

- Using the sqlCI.exe command line

Build step command line options
- /scriptsFolder
- /packageId
- /packageVersion
- /temporaryDatabaseServer
- /temporaryDatabaseName (optional)
- /temporaryDatabasePassword (optional)
- /outputFolder (optional)
- /additionalCompareArgs (optional)
- /licenseSerialKey (optional)

Test step command line options
- /package
- /temporaryDatabaseServer
- /temporaryDatabaseName (optional)
- /temporaryDatabaseUserName (optional)
- /temporaryDatabasePassword (optional)
- /outputFolder (optional)
- /sqlDataGenerator (optional)
- /additionalCompareArgs (optional)
- /licenseSerialKey (optional)
- /runOnly (optional)

Sync step command line options
- /databaseName
- /package
- /databaseServer
- /databaseUserName (optional)
- /databasePassword (optional)
- /additionalCompareArgs (optional)
- /licenseSerialKey (optional)

Publish step command line options
- /package
- /nugetFeedUrl

Activate command line option

Using the sqlCI.exe command line

The command line options for each step are case sensitive, but there's no restriction on the number of options you can use or the order in which you use them.

Options can be preceded by a forward slash '/' or two dashes '--'. In the examples on this page, we've used '/'.

Build step command line options

To run the build step, use this syntax:

```
Build /scriptsFolder=<value1> /packageId=<value2> /packageVersion=<value3>
/temporaryDatabaseServer=<value4>
```

Example

```
Build /scriptsFolder=WidgetShop\Database\Scriptsfolder
/packageId=WidgetShop /packageVersion=1.0 /temporaryDatabaseServer=
(localdb)\v11.0
```

The rest of this section provides examples of each build option.

/scriptsFolder

The path to the database scripts folder in source control.

Example

```
/scriptsFolder="WidgetShop\Database\Scriptsfolder"
```

For the build VCS root, use a period (.):

```
/scriptsFolder=.
```

/packageId

The name of the NuGet package you're creating.

Example

```
/packageId=WidgetShop
```

The name must not contain spaces.

/packageVersion

The package build number.

Example

```
/packageVersion=1.0
```

If you're running sqlCI.exe from the command line, you have to specify this number manually, but if you're running sqlCI.exe as part of an automated build, you can use the build number variable. To set the build number variable:

- in TeamCity, enter `$(build.number)`
- in CruiseControl.NET, enter `$(CCNetLabel)`
- in TFS 2008 or earlier, enter `($BuildNumber)`
- in TFS 2010 or later, follow the instructions in Passing TFS Build properties to MSBuild using TFS 2013, 2012 and 2010
- in Bamboo, enter `${bamboo.buildNumber}`

If you're using a different build system, check the documentation provided by the vendor.

/temporaryDatabaseServer

You don't need to configure this option if you're using SQL Automation Pack 1.1.0.1840 and later, and you have LocalDB installed. SQL CI will use LocalDB by default.

You do need to configure it if you want to run full-text queries against your tables. LocalDB doesn't support Full-Text Search, so you should use a temporary database on your instance of SQL Server instead. For more information on LocalDB, see SQL Server 2014 Express Local DB (MSDN article).

The temporary database server name.

Example

```
/temporaryDatabaseServer=SQLServer2012
```

During the build step, SQL CI needs to recreate your database on a temporary server. We recommend using localDB for this.

/temporaryDatabaseName (optional)

You should only use this option if:

- you don't have permissions to create databases on your SQL Server.
- you don't plan to run multiple builds in parallel that use the same temporary database. You must use a separate database for each build definition.

If you don't use this option, SQL CI will create a temporary database by default.

This option is only available if you're using SQL Automation Pack v1.1.1.1950 (includes SQL CI v2.0.1.315).

The name of an existing database on the temporary database server you previously specified.

Example
`/temporaryDatabaseName=WidgetShopTemp`

During the build step, SQL CI needs to recreate your database on a temporary server. You should use an empty database for this, because data will be overwritten during the build and test steps.

/temporaryDatabaseUserName (optional)

The username for SQL authentication.

Example
`/temporaryDatabaseUserName=sa`

If you don't specify this option, Windows authentication is used by default.

/temporaryDatabasePassword (optional)

The password for SQL authentication.

Example
`/temporaryDatabasePassword="password"`

This option is required if you're using the `/temporaryDatabaseUserName` option.

/outputFolder (optional)

The output folder path.

Example
`/outputFolder="c:\temp"`

If you don't specify this option, the current working directory is used by default. Make sure you have permission to create new files and folders in this directory.

/additionalCompareArgs (optional)

To include SQL Compare command line options use `="/options:<option_name>" or ="/options:+<option_name>"`.

Example

```
/additionalCompareArgs="/options:DoNotOutputCommentHeader,ForceColumnOrder"
```

The options are not case-sensitive.

You can also use aliases if you prefer shorthand versions of the commands. For details of the alias for each command, see Options used in the command line.

Example

```
/additionalCompareArgs="/options:nc,f"
```

To exclude default SQL Compare options, use `="/options:-<option_name>"`.

Example

```
/additionalCompareArgs="/options:-IncludeDependencies,-IgnoreWhiteSpace"
```

The default options (with aliases in brackets) included in the build step are:

- DecryptPost2KEncryptedObjects (dp2k)
- IgnoreFillFactor (if)
- IgnoreWhiteSpace (iw)
- IncludeDependencies (incd)
- IgnoreFileGroups (ifg)
- IgnoreUserProperties (iup)
- IgnoreWithElementOrder (iweo)
- IgnoreDatabaseAndServerName (idsn)

You can include some options and exclude others in the same command.

Example

```
/additionalCompareArgs="/options:IgnoreConstraintNames,NoTransactions,-IgnoreFillFactor"
```

To include SQL Compare command line switches:

```
/additionalCompareArgs="/AbortOnWarnings:[None]/[High]/[Medium]"
```

If the SQL Compare command line option or switch contains a space, you need to enclose the command in double quotes and insert escape characters before each quote.

Example

```
/additionalCompareArgs="/TIL:\"READ COMMITTED\""
```

If you don't escape the double quotes, the option will be invalid.

For more information about SQL Compare command line options and switches, see Using the command line.

/licenseSerialKey (optional)

To activate the SQL Automation Pack license on the machine on which the build agent is running, enter the serial number.

Example

```
/licenseSerialKey=123-456-789012-ABCD
```

For information on finding your Automation Pack serial number, see Licensing. If you don't enter a serial number, a 28 day free trial of the SQL Automation Pack will start automatically (14 day trial if you're using SQL Automation Pack v1.1.0.1840 and earlier).

If you have multiple serial numbers, separate them using commas without spaces.

Example
`/licenseSerialKey=123-456-789012-ABCD,321-456-987654-DCBA`

Test step command line options

> If you want to run tSQLt tests against the package, you'll need to enable the common language runtime (CLR) integration feature on the /temporaryDatabaseServer. For more information, see Enabling CLR Integration (TechNet article).

To run the test step, use this syntax:

```
Test /package=<value1> /temporaryDatabaseServer=<value2>
```

Example
`Test /package=WidgetShop\Database\Output\WidgetShop.1.0.nupkg /temporaryDatabaseServer=(localdb)\v11.0`

The rest of this section provides examples of each test option.

/package

The path to the package that was created in the build step.

Example
`/package=WidgetShop\Database\Output\WidgetShop.1.0.nupkg`

/temporaryDatabaseServer

> You don't need to configure this option if you're using SQL Automation Pack 1.1.0.1840 and later, and you have LocalDB installed. SQL CI will use LocalDB by default.
>
> You do need to configure it if you want to run full-text queries against your tables. LocalDB doesn't support Full-Text Search, so you should use a temporary database on your instance of SQL Server instead. For more information on LocalDB, see SQL Server 2014 Express Local DB (MSDN article).

The temporary database server name.

Example
`/temporaryDatabaseServer=SQLServer2012`

During the build step, SQLCI recreates your database on a temporary server. During the test step, tSQLt tests are run against it. We recommend using localDB for this.

/temporaryDatabaseName (optional)

You should only use this option if:

- you don't have permissions to create databases on your SQL Server.
- you don't plan to run multiple builds in parallel that use the same temporary database. You must use a separate database for each build definition.

If you don't use this option, SQL CI will create a temporary database by default.

This option is only available if you're using SQL Automation Pack v1.1.1.1950 (includes SQL CI v2.0.1.315).

The name of an existing database on the temporary database server you previously specified.

Example
`/temporaryDatabaseName=WidgetShopTemp`

During the build step, SQL CI recreates your database on a temporary server. During the test step, tSQLt are run against it.

/temporaryDatabaseUserName (optional)

The username for SQL authentication.

Example
`/temporaryDatabaseUserName=sa`

If you don't specify this option, Windows authentication is used by default.

/temporaryDatabasePassword (optional)

The password for SQL authentication.

Example
`/temporaryDatabasePassword="password"`

/outputFolder (optional)

The output folder for test results.

Example
`/outputFolder="c:\testresults"`

If you don't specify this option, the current working directory is used by default.

/sqlDataGenerator (optional)

Use test data created by SQL Data Generator. If you've already created a SQL Data Generator project file, include the file name.

Example
`/sqlDataGenerator="TestData.sqlgen"`

If you don't have an existing project file, include this option without specifying a value. This will populate the temporary database with test data that's been generated automatically. For more information about SQL Data Generator, see SQL Data Generator 3 documentation.

/additionalCompareArgs (optional)

To include SQL Compare command line options use `="/options:<option_name>" or ="/options:+<option_name>"`.

Example

```
/additionalCompareArgs="/options:DoNotOutputCommentHeader,ForceColumnOrder"
```

The options are not case-sensitive.

You can also use aliases if you prefer shorthand versions of the commands. For details of the alias for each command, see Options used in the command line.

Example

```
/additionalCompareArgs="/options:nc,f"
```

To exclude default SQL Compare options, use `="/options:-<option_name>"`.

Example

```
/additionalCompareArgs="/options:-IncludeDependencies,-IgnoreWhiteSpace"
```

The default options (with aliases in brackets) included in the build step are:

- DecryptPost2KEncryptedObjects (dp2k)
- IgnoreFillFactor (if)
- IgnoreWhiteSpace (iw)
- IncludeDependencies (incd)
- IgnoreFileGroups (ifg)
- IgnoreUserProperties (iup)
- IgnoreWithElementOrder (iweo)
- IgnoreDatabaseAndServerName (idsn)

You can include some options and exclude others in the same command.

Example

```
/additionalCompareArgs="/options:IgnoreConstraintNames,NoTransactions,-IgnoreFillFactor"
```

To include SQL Compare command line switches:

```
/additionalCompareArgs="/AbortOnWarnings:[None]/[High]/[Medium]"
```

If the SQL Compare command line option or switch contains a space, you need to enclose the command in double quotes and insert escape characters before each quote.

Example

```
/additionalCompareArgs="/TIL:\"READ COMMITTED\""
```

If you don't escape the double quotes, the option will be invalid.

For more information about SQL Compare command line options and switches, see Using the command line.

/licenseSerialKey (optional)

To activate the SQL Automation Pack license on the machine on which the build agent is running, enter the serial number.

Example

```
/licenseSerialKey=123-456-789012-ABCD
```

For information on finding your SQL Automation Pack serial number, see Licensing. If you don't enter a serial number, a 28 day free trial of the SQL Automation Pack will start automatically (14 day trial if you're using SQL Automation Pack v1.1.0.1840 and earlier).

If you have multiple serial numbers, separate them using commas without spaces.

Example

```
/licenseSerialKey=123-456-789012-ABCD,321-456-987654-DCBA
```

/runOnly (optional)

To run a specific test class or test instead of every test, enter the test class and test name.

Example

```
/runOnly=[testclass].[testname]
```

If you want to run every test, don't specify this option.

This command is only available in SQL Automation Pack 1.1.0.1840 and later.

Sync step command line options

To run the sync step, use this syntax:

```
sqlCI.exe Sync /databaseName=<value1> /package=<value2> /databaseServer=<value3>
```

Example

```
Sync /databaseName=WidgetShop /package=WidgetShop\Database\Output\WidgetShop.1.0.nupkg
/databaseServer=SQLServer2012
```

The rest of this section provides examples of each sync option.

/databaseName

The target database to update with the changes in source control. This must be an existing database on the server; the runner does not create the database for you.

Example

```
/databaseName=WidgetDB
```

/package

The path to the package that was created in the build step.

Example
`/package=WidgetShop\Database\Output\WidgetShop.1.0.nupkg`

/databaseServer

The target database server name.

Example
`/databaseServer=SQLServer2012`

/databaseUserName (optional)

The username for SQL authentication.

Example
`/databaseUserName=sa`

/databasePassword (optional)

The password for SQL authentication.

Example
`/databasePassword="password"`

/additionalCompareArgs (optional)

To include SQL Compare command line options use `="/options:<option_name>" or ="/options:+<option_name>"`.

Example
`/additionalCompareArgs="/options:DoNotOutputCommentHeader,ForceColumnOrder"`

The options are not case-sensitive.

You can also use aliases if you prefer shorthand versions of the commands. For details of the alias for each command, see Options used in the command line.

Example
`/additionalCompareArgs="/options:nc,f"`

To exclude default SQL Compare options, use `="/options:-<option_name>"`.

Example
`/additionalCompareArgs="/options:-IncludeDependencies,-IgnoreWhiteSpace"`

The default options (with aliases in brackets) included in the build step are:

- DecryptPost2KEncryptedObjects (dp2k)
- IgnoreFillFactor (if)

- IgnoreWhiteSpace (iw)
- IncludeDependencies (incd)
- IgnoreFileGroups (ifg)
- IgnoreUserProperties (iup)
- IgnoreWithElementOrder (iweo)
- IgnoreDatabaseAndServerName (idsn)
- IgnoretSQLt (itst)

If the target database you're syncing to contains tSQLt tests, you must exclude the IgnoretSQLt option.

You can include some options and exclude others in the same command.

Example

```
/additionalCompareArgs="/options:IgnoreConstraintNames,NoTransactions,-IgnoreFillFactor"
```

To include SQL Compare command line switches:

```
/additionalCompareArgs="/AbortOnWarnings:[None]/[High]/[Medium]"
```

If the SQL Compare command line option or switch contains a space, you need to enclose the command in double quotes and insert escape characters before each quote.

Example

```
/additionalCompareArgs="/TIL:\"READ COMMITTED\""
```

If you don't escape the double quotes, the option will be invalid.

For more information about SQL Compare command line options and switches, see Using the command line.

/licenseSerialKey (optional)

To activate the SQL Automation Pack license on the machine on which the build agent is running, enter the serial number.

Example

```
/licenseSerialKey=123-456-789012-ABCD
```

For information on finding your SQL Automation Pack serial number, see Licensing. If you don't enter a serial number, a 28 day free trial of the SQL Automation Pack will start automatically (14 day trial if you're using SQL Automation Pack v1.1.0.1840 and earlier).

If you have multiple serial numbers, separate them using commas without spaces.

Example

```
/licenseSerialKey=123-456-789012-ABCD,321-456-987654-DCBA
```

Publish step command line options

To run the publish step, use this syntax:

```
Publish /package=<value1> /nugetFeedUrl=<value2>
```

Example

```
Publish /package=WidgetShop\Database\Output\WidgetShop.1.0.nupkg /nugetFeedUrl=http://nugetfeedmachinename:8081/nuget/
```

The rest of this section provides examples of each publish option.

/package

The path to the package that was created in the build step.

Example

```
/package=WidgetShop\Database\Output\WidgetShop.1.0.nupkg
```

/nugetFeedUrl

The fully-qualified URL for your NuGet feed. If your are using Deployment Manager, this is the URL for Deployment Manager with `/nuget/` added to the end.

Example

```
/nugetFeedUrl=http://nugetfeedmachinename:8081/nuget/
```

/nugetFeedApiKey (optional)

The API key for your NuGet feed.

Example

```
/nugetFeedApiKey=DO8XW4CI7UIROVCFG4TYTC2DXM8
```

If you are using a public NuGet feed, the API key is not required.

Activate command line option

To activate the SQL Automation Pack license without having to configure a build, test or sync step, use this syntax:

```
Activate /licenseSerialKey=<value>
```

Example

```
Activate /licenseSerialKey=123-456-789012-ABCD
```

This also activates SQL CI.

For information on finding your SQL Automation Pack serial number, see Licensing.

If you have multiple serial numbers, separate them using commas without spaces.

Example
`Activate /licenseSerialKey=123-456-789012-ABCD,321-456-987654-DCBA`