

Apex SQL

Apex SQL Complete

ApexSQL Complete can automatically fill in SQL keywords, statements, and identifiers. It integrates seamlessly with SSMS and helps with object script and description reviews without coding interruption and also speeds up SQL coding with snippets. This free SSMS can quickly reference the object's script and description inline, format keywords and fill in fully qualified object names automatically.

[Apex SQL Complete Introduction](#)

With ApexSQL Complete you can:

- Auto-complete SQL code directly in SQL Server Management Studio, including SSMS 17
- Identify the structure of complex SQL queries at a glance
- Format auto-completed keywords in upper, lower or proper case
- Specify aliases to be filled in for objects from all SQL Server instances and databases
- Boost auto-complete performance with object caching

[More Information](#)

Install the Add-on

The add on can be found locally on Adriano. The following path can be used:

```
\\Adriano\FileServer\CDRack\SQL_Apex\ApexSQLInstaller.exe
```

The add on must be installed using the default path.

Configuring the signify Profile

Please note that SSMS must be closed before adding the configured files. In order to allow all user using the auto-complete add-on, a default signify profile has been configured. After the installation has been completed, the attached config package, ApexSQLComplete.zip, must be downloaded and copied to the user's local file directory.

Using the default installation path, the package must then be extracted to the following location:

```
C:\Users\[user]\AppData\Local\ApexSQL\ApexSQLComplete.
```

Within the files MyDefaults.xml and Options.xml replace the [user] with you domain user name to allow executed queries to be stored correctly to your computers file structure.

Once the files have been replaced in this folder all setup will be ready for use after SSMS has been reopened. The following tools are now available:

Auto Complete

To insert auto-complete code the following insertion keys are available:

- Space
- Tab

To show hints for typed code press **ctrl+space**

Execution Alert

In order to avoid dangerous updates from occurring on any DB, the following alerts have been configured to notify the user of such an action before continuing.

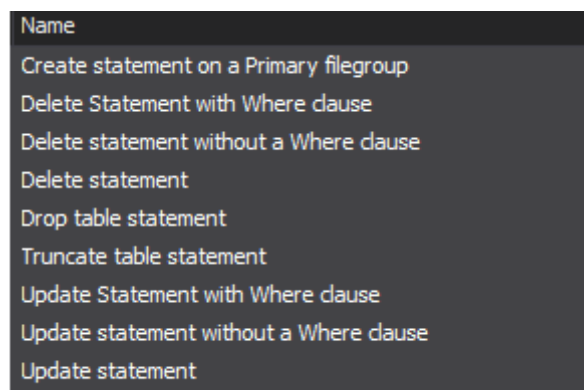


Figure 1: Execution Alerts

Snippets

This is prepared code that can be inserted directly into the query with the basic building blocks required by typing the snippet key and clicking it:

Table 1: SQL Snippets

Snippet key	Description
TableCreate	Creates a table with all the signify entity defaults columns added
FKCreate	Check for existence of foreign key and add if not exist as specified on Add Foreign Key Constraint page

Executed Query Log

This is a log of all queries executed on the user's SSMS and are stored for the last 7 days.

Tab Colouring

The addition of tab colouring allows the user to immediately confirm on what server a query will be executed and the risk level of the execution. The following colouring has been configured:

Server	Database	Environment	Color
cassio\sql2014	SignifyHR_MasterDev	MasterDev	
shakespeare\sql2008	<All databases>	MasterQA	
10.10.121.121	<All databases>	Production	
Quintus\Clientdatabases	<All databases>	Local_Clients	
54.194.247.91	<All databases>	Production	
cassio\sql2014	<All databases>	FeatureBranches	
shakespeare\sql2014	<All databases>	Local_Clients	
shakespeare\sql2008	MasterBuilder	MasterBuilder	

Figure 2: Server DB Tab Colouring

Auto Replace

The following auto replacements will be made for keys given followed by the space bar press

Table 2: Auto Replacement Keys

Auto Replacement Key	Replacement
loc	WITH(NOLOCK)
sel	SELECT * FROM
gdate	GETDATE() BETWEEN ValidFrom and ValidTo
orderb	ORDER BY id DESC
ProcExist	IF EXISTS (SELECT TOP 1 1 FROM sys.objects WHERE object_id = OBJECT_ID(N'myproc') AND type IN (N'P', N'PC')) DROP PROCEDURE myproc GO
TableExist	IF (EXISTS (SELECT TOP 1 1 FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'TheTable')) BEGIN /*Do Stuff*/ END
TempTableExist	IF OBJECT_ID('tempdb..#TheTable') IS NOT NULL BEGIN /*Do Stuff*/ END
ViewExist	IF EXISTS(SELECT TOP 1 1 FROM sys.views WHERE name = 'TheView') BEGIN /*Do Stuff*/ END

FunctionExist	IF EXISTS (SELECT TOP 1 1 FROM sys.objects WHERE object_id = OBJECT_ID(N'[dbo].[FunctionName]') AND type IN (N'FN', N'IF', N'TF', N'FS', N'FT')) BEGIN /* Do your stuff here */ END GO
IndexExist	IF EXISTS(SELECT TOP 1 1 FROM sys.indexes WHERE name = 'IndexName' AND object_id = OBJECT_ID('TableName')) BEGIN /* Do your stuff here */ END
DefaultExist	IF NOT EXISTS (SELECT TOP 1 1 FROM sys.all_columns c JOIN sys.tables t ON t.object_id = c.object_id JOIN sys.schemas s ON s.schema_id = t.schema_id JOIN sys.default_constraints d ON c.default_object_id = d.object_id WHERE t.name = 'TABLE_NAME' AND c.name = 'COLUMN_NAME' AND s.name = 'dbo') BEGIN /* Do your stuff here */ END
CommentChanged	{Date} : {Domain Username} : {Short description of change}
CommentCreated	{Date} : {Domain Username} : {Short description of change}
StatExec	Use SELECT execquery.last_execution_time AS [Date Time], execsql.text AS [Script] FROM sys.dm_exec_query_stats AS execquery CROSS APPLY sys.dm_exec_sql_text(execquery.sql_handle) AS execsql ORDER BY execquery.last_execution_time DESC
StatProc	SELECT TOP 1000 d.object_id, d.database_id, OBJECT_NAME(object_id, database_id) 'proc name', d.cached_time, d.last_execution_time, d.total_elapsed_time, d.total_elapsed_time/d.execution_count AS [avg_elapsed_time], d.last_elapsed_time, d.execution_count ,d.last_elapsed_time/(1024*1024) as secondes, [total_worker_time] FROM sys.dm_exec_procedure_stats AS d ORDER BY d.last_elapsed_time desc, [total_worker_time] DESC;
StatSession	SELECT r.start_time [Start Time],session_ID [SPID], DB_NAME(database_id) [Database], SUBSTRING(t.text,(r.statement_start_offset/2)+1, CASE WHEN statement_end_offset=-1 OR statement_end_offset=0 THEN (DATALENGTH(t.Text)-r.statement_start_offset/2)+1 ELSE (r.statement_end_offset-r.statement_start_offset)/2+1 END) [Executing SQL], Status,command,wait_type,wait_time,wait_resource, last_wait_type FROM sys.dm_exec_requests r OUTER APPLY sys.dm_exec_sql_text(sql_handle) t WHERE session_id != @@SPID don't show this query AND session_id > 50 don't show system queries ORDER BY r.start_time

StatHealth	<pre> SELECT TOP 10000 record_id ,dateadd(ms, - 1 * (@ms_ticks_now - [timestamp])), GetDate()) AS EventTime ,SQLProcessUtilization ,SystemIdle ,100 - SystemIdle - SQLProcessUtilization AS OtherProcessUtilization FROM (SELECT record.value('(/Record/@id)[1]', 'int') AS record_id ,record.value('(/Record/SchedulerMonitorEvent/SystemHealth/SystemIdle)[1]', 'int') AS SystemIdle ,record.value('(/Record/SchedulerMonitorEvent/SystemHealth/ProcessUtilization)[1]', 'int') AS SQLProcessUtilization ,TIMESTAMP FROM (SELECT TIMESTAMP ,convert(XML, record) AS record FROM sys.dm_os_ring_buffers WHERE ring_buffer_type = N'RING_BUFFER_SCHEDULER_MONITOR' AND record LIKE '%%') AS x) AS y ORDER BY record_id DESC </pre>
KillDB	<pre> USE master GO DECLARE @kill varchar(8000) = ; SELECT @kill = @kill + 'kill ' + CONVERT(varchar(5), spid) + ';' FROM master..sysprocesses WHERE dbid = db_id('SignifyHR_ARMUAT_CW_Import_NVE') EXEC(@kill); ALTER DATABASE SignifyHR_ARMUAT_CW_Import_NVE SET MULTI_USER WITH ROLLBACK IMMEDIATE GO </pre>
ColumnExist	<pre> IF EXISTS(SELECT TOP 1 1 FROM sys.columns WHERE Name = N'columnName' AND Object_ID = Object_ID(N'schemaName.tableName')) BEGIN Column Exists END </pre>
ConstraintExist	<pre> IF EXISTS (SELECT TOP 1 1 FROM SYS.DEFAULT_CONSTRAINTS WHERE OBJECT_NAME(PARENT_OBJECT_ID) = 'RequisitionExternalCandidates' AND COL_NAME(PARENT_OBJECT_ID, PARENT_COLUMN_ID) = 'CreatedDate') BEGIN /*Do Stuff*/ END </pre>
ScriptName	<pre> exec GenerateSQLFileName 'DomainUserName', 'ObjectName', 'SQLType', /*(Optional)*/'CommaDelimitedUsedByScripts' </pre>
RowNum	<pre> ROW_NUMBER() OVER(PARTITION BY TerritoryName ORDER BY SalesYTD DESC) </pre>
ForeignKeyExist	<pre> IF NOT EXISTS (SELECT 1 FROM sys.foreign_keys AS f INNER JOIN sys.foreign_key_columns AS fc ON f.OBJECT_ID = fc.constraint_object_id INNER JOIN sys.tables t ON t.OBJECT_ID = fc.referenced_object_id WHERE OBJECT_NAME (f.referenced_object_id) = 'licLicenceReasons' AND OBJECT_NAME(f.parent_object_id) = 'prsEmployeeLicences' AND COL_NAME(fc.parent_object_id,fc.parent_column_id) = 'licLicenceReasonId') ALTER TABLE prsEmployeeLicences ADD CONSTRAINT FK_prsEmployeeLicences_licLicenceReasons FOREIGN KEY (licLicenceReasonId) REFERENCES licLicenceReasons(Id) GO </pre>

	<pre> /* C = CHECK constraint D = Default or DEFAULT constraint F = FOREIGN KEY constraint L = Log FN = Scalar function IF = Inlined table-function P = Stored procedure PK = PRIMARY KEY constraint (type is K) RF = Replication filter stored procedure S = System table TF = Table function TR = Trigger U = User table UQ = UNIQUE constraint (type is K) V = View X = Extended stored procedure • / IF (OBJECT_ID('tempdb..#tmpjplImportUpdateData', 'U') IS NOT NULL) BEGIN END </pre>
ObjectExist	<pre> DECLARE @SchemaID INT DECLARE curs CURSOR FAST_FORWARD FOR SELECT SchemaID FROM cfgSchemaID WITH(NOLOCK) WHERE GETDATE() BETWEEN ValidFrom AND ValidTo AND SysID = 101 OPEN curs FETCH NEXT FROM curs INTO @SchemaID WHILE @@FETCH_STATUS = 0 BEGIN /*Do your commands for @SchemaID here*/ /*Get the next author.*/ FETCH NEXT FROM curs INTO @SchemaID END CLOSE curs DEALLOCATE curs </pre>
SchemaCursor	

Apex SQL Refactor

ApexSQL Refactor is a SQL Server Management Studio and Visual Studio free add-in, for format and refactor SQL code and objects. ApexSQL Refactor has 200+ formatting options and nearly 15 code refactors

Apex SQL Refactor Introduction

ApexSQL Refactor can:

- Qualify SQL Server object names, expand wild cards and locate and highlight unused variables and parameters

- Encapsulate SQL code into procedures, replace one-to-many relationships, add surrogate keys, change procedure parameters
- Update all dependent database objects for renaming or changing columns and parameters without breaking any dependencies
- Format SQL code in the SSMS or Visual Studio query window, it can format SQL objects or external SQL scripts

[More Information](#)

Install the Add-on

The add-on can be found locally on Adriano. The following path can be used:

```
\\Adriano\FileServer\CDRack\SQL_Apex\Apex SQL Refactor\ApexSQLRefactor.exe
```

Configuring the signify Profile

In order to allow all user using the refactor add-on a default in signify profile has been configured. After the installation has been completed the attached refactor profile, FormattingSettings.zip, must be downloaded and imported in SSMS using the following path:

```
ApexSQL|Apex SQL Refactor|Options|Import
```

Once the import has been completed, select the Signify profile under the profile drop down list and set the profile as active. The window can now be closed. To refactor a query using this tool press **ctrl+shift+alt+F**

In order to replace * in select * from with the corresponding column names press **Ctrl+Shift+Alt+U**

Apex SQL Search

ApexSQL Search is a free SQL search add-in for SSMS and Visual Studio. It offers text search in SQL database objects and data, allows safe renaming of SQL objects, and graphical visualization of object interdependencies

[Apex SQL Search Introduction](#)

With ApexSQL Search you can:

- Search for text within database objects (including object names) using the Object search
- Search for data stored in tables and views (even encrypted ones) using the Text search
- Repeat previous searches in a single click using the Search history
- Visualize (and print) all objects' relationships using the Dependency viewer

- Change the name and schema of tables, views, stored procedures, functions, columns, and parameters without breaking your databases with the Safe rename refactor
- Visually edit SQL objects' extended properties via the Extended property editor feature

[More Information](#)

Install the Add-on

The add-on can be found locally on Adriano. The following path can be used:

```
\\Adriano\FileServer\CDRack\SQL_Apex\Apex SQL Search\ApexSQLSearch.exe
```

Revision #9

Created 26 May 2020 17:07:39 by Nardus van Eyk

Updated 10 February 2022 07:56:06