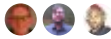






SQL Server Utilities Statements - GO

2017-3-14 • 2 min to read • Contributors 

In this article

- Syntax
- Arguments
- Remarks
- Permissions
- Examples

THIS TOPIC APPLIES TO:  SQL Server (starting with 2008)  Azure SQL Database  Azure SQL Data Warehouse  Parallel Data Warehouse


SQL Server provides commands that are not Transact-SQL statements, but are recognized by the **sqlcmd** and **osql** utilities and SQL Server Management Studio Code Editor. These commands can be used to facilitate the readability and execution of batches and scripts.

GO signals the end of a batch of Transact-SQL statements to the SQL Server utilities. +

```
||
|-|
|Applies to: SQL Server ( SQL Server 2008 through current version), Azure SQL Database.|
```

 [Transact-SQL Syntax Conventions](#)

Syntax

	 Copy
GO [count]	

Arguments

count
Is a positive integer. The batch preceding GO will execute the specified number of times.

Remarks

GO is not a Transact-SQL statement; it is a command recognized by the **sqlcmd** and **osql** utilities and SQL Server Management Studio Code editor.

SQL Server utilities interpret GO as a signal that they should send the current batch of Transact-SQL statements to an instance of SQL Server. The current batch of statements is composed of all statements entered since the last GO, or since the start of the ad hoc session or script if this is the first GO.

A Transact-SQL statement cannot occupy the same line as a GO command. However, the line can contain comments.

Users must follow the rules for batches. For example, any execution of a stored procedure after the first statement in a batch must include the EXECUTE keyword. The scope of local (user-defined) variables is limited to a batch, and cannot be referenced after a GO command.



```
USE AdventureWorks2012;
GO
DECLARE @MyMsg VARCHAR(50)
SELECT @MyMsg = 'Hello, World.'
GO -- @MyMsg is not valid after this GO ends the batch.

-- Yields an error because @MyMsg not declared in this batch.
PRINT @MyMsg
GO

SELECT @@VERSION;
-- Yields an error: Must be EXEC sp_who if not first statement in
-- batch.
sp_who
GO
```

SQL Server applications can send multiple Transact-SQL statements to an instance of SQL Server for execution as a batch. The statements in the batch are then compiled into a single execution plan. Programmers executing ad hoc statements in the SQL Server utilities, or building scripts of Transact-SQL statements to run through the SQL Server utilities, use GO to signal the end of a batch.

Applications based on the ODBC or OLE DB APIs receive a syntax error if they try to execute a GO command. The SQL Server utilities never send a GO command to the server.

Do not use a semicolon as a statement terminator after GO.

Permissions

GO is a utility command that requires no permissions. It can be executed by any user.



```
-- Yields an error because ; is not permitted after GO  
SELECT @@VERSION;  
GO;
```

Examples

The following example creates two batches. The first batch contains only a

`USE`AdventureWorks2012` statement to set the database context. The remaining statements use a local variable. Therefore, all local variable declarations must be grouped in a single batch. This is done by not having a `GO` command until after the last statement that references the variable.



```
USE AdventureWorks2012;  
GO  
DECLARE @NmbrPeople int  
SELECT @NmbrPeople = COUNT(*)  
FROM Person.Person;  
PRINT 'The number of people as of ' +  
      CAST(GETDATE() AS char(20)) + ' is ' +  
      CAST(@NmbrPeople AS char (10));  
GO
```

The following example executes the statements in the batch twice.



```
SELECT DB_NAME();  
SELECT USER_NAME();  
GO 2
```