**Stored Procedures (Database Engine)**

**SQL Server 2012**

[Other Versions](javascript:;)

Description: http://i.technet.microsoft.com/Areas/Epx/Content/Images/ImageSprite.png

* [SQL Server 2008 R2](http://technet.microsoft.com/en-us/library/ms190782(d=printer,v=sql.105).aspx)
* [SQL Server 2008](http://technet.microsoft.com/en-us/library/ms190782(d=printer,v=sql.100).aspx)
* [SQL Server 2005](http://technet.microsoft.com/en-us/library/ms190782(d=printer,v=sql.90).aspx)

A stored procedure in SQL Server is a group of one or more Transact-SQL statements or a reference to a Microsoft .NET Framework common runtime language (CLR) method. Procedures resemble constructs in other programming languages because they can:

* Accept input parameters and return multiple values in the form of output parameters to the calling program.
* Contain programming statements that perform operations in the database. These include calling other procedures.
* Return a status value to a calling program to indicate success or failure (and the reason for failure).

[Benefits of Using Stored Procedures](javascript:void(0))

The following list describes some benefits of using procedures.

Reduced server/client network traffic

The commands in a procedure are executed as a single batch of code. This can significantly reduce network traffic between the server and client because only the call to execute the procedure is sent across the network. Without the code encapsulation provided by a procedure, every individual line of code would have to cross the network.

Stronger security

Multiple users and client programs can perform operations on underlying database objects through a procedure, even if the users and programs do not have direct permissions on those underlying objects. The procedure controls what processes and activities are performed and protects the underlying database objects. This eliminates the requirement to grant permissions at the individual object level and simplifies the security layers.

The [EXECUTE AS](http://technet.microsoft.com/en-us/library/ms188354.aspx) clause can be specified in the CREATE PROCEDURE statement to enable impersonating another user, or enable users or applications to perform certain database activities without needing direct permissions on the underlying objects and commands. For example, some actions such as TRUNCATE TABLE, do not have grantable permissions. To execute TRUNCATE TABLE, the user must have ALTER permissions on the specified table. Granting a user ALTER permissions on a table may not be ideal because the user will effectively have permissions well beyond the ability to truncate a table. By incorporating the TRUNCATE TABLE statement in a module and specifying that module execute as a user who has permissions to modify the table, you can extend the permissions to truncate the table to the user that you grant EXECUTE permissions on the module.

When calling a procedure over the network, only the call to execute the procedure is visible. Therefore, malicious users cannot see table and database object names, embed Transact-SQL statements of their own, or search for critical data.

Using procedure parameters helps guard against SQL injection attacks. Since parameter input is treated as a literal value and not as executable code, it is more difficult for an attacker to insert a command into the Transact-SQL statement(s) inside the procedure and compromise security.

Procedures can be encrypted, helping to obfuscate the source code. For more information, see [SQL Server Encryption](http://technet.microsoft.com/en-us/library/bb510663.aspx).

Reuse of code

The code for any repetitious database operation is the perfect candidate for encapsulation in procedures. This eliminates needless rewrites of the same code, decreases code inconsistency, and allows the code to be accessed and executed by any user or application possessing the necessary permissions.

Easier maintenance

When client applications call procedures and keep database operations in the data tier, only the procedures must be updated for any changes in the underlying database. The application tier remains separate and does not have to know how about any changes to database layouts, relationships, or processes.

Improved performance

By default, a procedure compiles the first time it is executed and creates an execution plan that is reused for subsequent executions. Since the query processor does not have to create a new plan, it typically takes less time to process the procedure.

If there has been significant change to the tables or data referenced by the procedure, the precompiled plan may actually cause the procedure to perform slower. In this case, recompiling the procedure and forcing a new execution plan can improve performance.

[Types of Stored Procedures](javascript:void(0))

User-defined

A user-defined procedure can be created in a user-defined database or in all system databases except the **Resource** database. The procedure can be developed in either Transact-SQL or as a reference to a Microsoft .NET Framework common runtime language (CLR) method.

Temporary

Temporary procedures are a form of user-defined procedures. The temporary procedures are like a permanent procedure, except temporary procedures are stored in **tempdb**. There are two types of temporary procedures: local and global. They differ from each other in their names, their visibility, and their availability. Local temporary procedures have a single number sign (#) as the first character of their names; they are visible only to the current user connection, and they are deleted when the connection is closed. Global temporary procedures have two number signs (##) as the first two characters of their names; they are visible to any user after they are created, and they are deleted at the end of the last session using the procedure.

System

System procedures are included with SQL Server. They are physically stored in the internal, hidden **Resource** database and logically appear in the **sys** schema of every system- and user-defined database. In addition, the **msdb** database also contains system stored procedures in the **dbo** schema that are used for scheduling alerts and jobs. Because system procedures start with the prefix **sp\_**, we recommend that you do not use this prefix when naming user-defined procedures. For a complete list of system procedures, see [System Stored Procedures (Transact-SQL)](http://technet.microsoft.com/en-us/library/ms187961.aspx)

SQL Server supports the system procedures that provide an interface from SQL Server to external programs for various maintenance activities. These extended procedures use the xp\_ prefix. For a complete list of extended procedures, see [General Extended Stored Procedures (Transact-SQL)](http://technet.microsoft.com/en-us/library/ms187389.aspx).

Extended User-Defined

Extended procedures enable creating external routines in a programming language such as C. These procedures are DLLs that an instance of SQL Server can dynamically load and run.

|  |
| --- |
| **Note** |
| Extended stored procedures will be removed in a future version of SQL Server. Do not use this feature in new development work, and modify applications that currently use this feature as soon as possible. Create CLR procedures instead. This method provides a more robust and secure alternative to writing extended procedures. |

[Related Tasks](javascript:void(0))

|  |  |
| --- | --- |
| **Task Description** | **Topic** |
| Describes how to create a stored procedure. | [Create a Stored Procedure](http://technet.microsoft.com/en-us/library/ms345415.aspx) |
| Describes how to modify a stored procedure. | [Modify a Stored Procedure](http://technet.microsoft.com/en-us/library/ms345356.aspx) |
| Describes how to delete a stored procedure. | [Delete a Stored Procedure](http://technet.microsoft.com/en-us/library/ms345358.aspx) |
| Describes how to execute a stored procedure. | [Execute a Stored Procedure](http://technet.microsoft.com/en-us/library/ms189915.aspx) |
| Describes how to grant permissions on a stored procedure. | [Grant Permissions on a Stored Procedure](http://technet.microsoft.com/en-us/library/ms345484.aspx) |
| Describes how to return data from a stored procedure to an application. | [Return Data from a Stored Procedure](http://technet.microsoft.com/en-us/library/ms188655.aspx) |
| Describes how to recompile a stored procedure. | [Recompile a Stored Procedure](http://technet.microsoft.com/en-us/library/ms190439.aspx) |
| Describes how to rename a stored procedure. | [Rename a Stored Procedure](http://technet.microsoft.com/en-us/library/ms345394.aspx) |
| Describes how to view the definition of a stored procedure. | [View the Definition of a Stored Procedure](http://technet.microsoft.com/en-us/library/ms345443.aspx) |
| Describes how to view the dependencies on a stored procedure. | [View the Dependencies of a Stored Procedure](http://technet.microsoft.com/en-us/library/ms345404.aspx) |