CREATE USER (Transact-SQL)

Updated: August 4, 2015

Applies To: SQL Server 2014, SQL Server 2016 Preview

Adds a user to the current database. There are eleven types of users:

Users based on logins in master This is the most common type of user.

- User based on a login based on a Windows user.
- User based on a login based on a Windows group.
- User based on a login using SQL Server authentication.

Users that authenticate at the database Only allowed in a contained database.

- User based on a Windows user that has no login.
- User based on a Windows group that has no login.
- Contained database user with password.

Users based on Windows principals that connect through Windows group logins

- User based on a Windows user that has no login, but can connect to the Database Engine through membership in a Windows group.
- User based on a Windows group that has no login, but can connect to the Database Engine through membership in a different Windows group.

Users that cannot authenticate These users cannot login to SQL Server or SQL Database.

- User without a login. Cannot login but can be granted permissions.
- User based on a certificate. Cannot login but can be granted permissions and can sign modules.
- User based on an asymmetric key. Cannot login but can be granted permissions and can sign modules.

Applies to: SQL Server (SQL Server 2008 through current version), Azure SQL Database, SQL Database V12, (Preview in some regions), Azure SQL Data Warehouse Public Preview.

Transact-SQL Syntax Conventions

Syntax

```
-- SQL Server SyntaxUsers based on logins in master
CREATE USER user_name
    [
        { FOR | FROM } LOGIN login_name
    ]
    [ WITH DEFAULT_SCHEMA = schema_name ]
[ ; ]

Users that authenticate at the database
CREATE USER
    {
      windows_principal [ WITH <options_list> [ ,... ] ]

    | user_name WITH PASSWORD = 'password' [ , <options_list> [ ,... ]
    }
 [ ; ]

Users based on Windows principals that connect through Windows group logins
CREATE USER
    {
          windows_principal [ { FOR | FROM } LOGIN windows_principal ]
        | user_name { FOR | FROM } LOGIN windows_principal
    }
    [ WITH DEFAULT_SCHEMA = schema_name ]
[ ; ]

Users that cannot authenticate
CREATE USER user_name
    {
         WITHOUT LOGIN [ WITH DEFAULT_SCHEMA = schema_name ]
       | { FOR | FROM } CERTIFICATE cert_name
       | { FOR | FROM } ASYMMETRIC KEY asym_key_name
    }
 [ ; ]

<options_list> ::=
      DEFAULT_SCHEMA = schema_name
    | DEFAULT_LANGUAGE = { NONE | lcid | language name | language alias }
    | SID = sid
```

```
-- Windows Azure SQL Database
CREATE USER user_name
    [ { { FOR | FROM }
      {
        LOGIN login_name
      }
      | WITHOUT LOGIN
      }
    ]
    [ WITH DEFAULT_SCHEMA = schema_name ]
[;]
-- SQL Database syntax when connected to a federation member

CREATE USER user_name
[;]
```

Arguments

user_name

Specifies the name by which the user is identified inside this database. *user_name* is a **sysname**. It can be up to 128 characters long. When creating a user based on a Windows principal, the Windows principal name becomes the user name unless another user name is specified.

LOGIN *login_name*

Specifies the login for which the database user is being created. *login_name* must be a valid login in the server. Can be a login based on a Windows principal (user or group), or a login using SQL Server authentication. When this SQL Server login enters the database, it acquires the name and ID of the database user that is being created. When creating a login mapped from a Windows principal, use the format **[**<*domainName*>****<*loginName*>**]**. For examples, see Syntax Summary.

If the CREATE USER statement is the only statement in a SQL batch, Windows Azure SQL Database supports the WITH LOGIN clause. If the CREATE USER statement is not the only statement in a SQL batch or is executed in dynamic SQL, the WITH LOGIN clause is not supported.

WITH DEFAULT_SCHEMA = *schema_name*

Specifies the first schema that will be searched by the server when it resolves the names of objects for this database user.

'*windows_principal*'

Specifies the Windows principal for which the database user is being created. The *windows_principal* can be a Windows user, or a Windows group. The user will be created even if the *windows_principal* does not have a login. When connecting to SQL Server, if the *windows_principal* does not have a login, the Windows principal must authenticate at the Database Engine through membership in a Windows group that has a login, or the connection string must specify the contained database as the initial catalog. When creating a user from a Windows principal, use the format **[**<*domainName*>****<*loginName*>**]**. For examples, see Syntax Summary.

WITH PASSWORD = '*password*'

Applies to: SQL Server 2012 through SQL Server 2016, SQL Database V12.

Can only be used in a contained database. Specifies the password for the user that is being created.

WITHOUT LOGIN

Specifies that the user should not be mapped to an existing login.

CERTIFICATE *cert_name*

Applies to: SQL Server 2008 through SQL Server 2016, SQL Database V12.

Specifies the certificate for which the database user is being created.

ASYMMETRIC KEY *asym_key_name*

Applies to: SQL Server 2008 through SQL Server 2016, SQL Database V12.

Specifies the asymmetric key for which the database user is being created.

DEFAULT_LANGUAGE = *{ NONE* | *<lcid>* | *<language name>* | *<language alias> }*

Applies to: SQL Server 2012 through SQL Server 2016, SQL Database V12.

Specifies the default language for the new user. If a default language is specified for the user and the default language of the database is later changed, the users default language remains as specified. If no default language is specified, the default language for the user will be the default language of the database. If the default language for the user is not specified and the default language of the database is later changed, the default language of the user will change to the new default language for the database.

Important

DEFAULT_LANGUAGE is used only for a contained database user.

SID = *sid*

Applies to: SQL Server 2012 through SQL Server 2016.

Applies only to users with passwords (SQL Server authentication) in a contained database. Specifies the SID of the new database user. If this option is not selected, SQL Server automatically assigns a SID. Use the SID parameter to create users in multiple databases that have the same identity (SID). This is useful when creating users in multiple databases to prepare for AlwaysOn failover. To determine the SID of a user, query sys.database_principals.

Remarks

If FOR LOGIN is omitted, the new database user will be mapped to the SQL Server login with the same name.

The default schema will be the first schema that will be searched by the server when it resolves the names of objects for this database user. Unless otherwise specified, the default schema will be the owner of objects created by this database user.

If the user has a default schema, that default schema will used. If the user does not have a default schema, but the user is a member of a group that has a default schema, the default schema of the group will be used. If the user does not have a default schema, and is a member of more than one group, the default schema for the user will be that of the Windows group with the lowest principal_id and an explicitly set default schema. (It is not possible to explicitly select one of the available default schemas as the preferred schema.) If no default schema can be determined for a user, the **dbo** schema will be used.

DEFAULT_SCHEMA can be set before the schema that it points to is created.

DEFAULT_SCHEMA cannot be specified when you are creating a user mapped to a certificate, or an asymmetric key.

The value of DEFAULT_SCHEMA is ignored if the user is a member of the sysadmin fixed server role. All members of the sysadmin fixed server role have a default schema of dbo.

The WITHOUT LOGIN clause creates a user that is not mapped to a SQL Server login. It can connect to other databases as guest. Permissions can be assigned to this user without login and when the security context is changed to a user without login, the original users receives the permissions of the user without login. See example D. Creating and using a user without a login.

Only users that are mapped to Windows principals can contain the backslash character (\).

CREATE USER cannot be used to create a guest user because the guest user already exists inside every database. You can enable the guest user by granting it CONNECT permission, as shown:

```
GRANT CONNECT TO guest;
GO
```

Information about database users is visible in the sys.database_principals catalog view.

Syntax Summary

Users based on logins in master

The following list shows possible syntax for users based on logins. The default schema options are not listed.

- `CREATE USER [Domain1\WindowsUserBarry]`
- `CREATE USER [Domain1\WindowsUserBarry] FOR LOGIN Domain1\WindowsUserBarry`
- `CREATE USER [Domain1\WindowsUserBarry] FROM LOGIN Domain1\WindowsUserBarry`
- `CREATE USER [Domain1\WindowsGroupManagers]`
- `CREATE USER [Domain1\WindowsGroupManagers] FOR LOGIN [Domain1\WindowsGroupManagers]`
- `CREATE USER [Domain1\WindowsGroupManagers] FROM LOGIN [Domain1\WindowsGroupManagers]`
- `CREATE USER SQLAUTHLOGIN`
- `CREATE USER SQLAUTHLOGIN FOR LOGIN SQLAUTHLOGIN`
- `CREATE USER SQLAUTHLOGIN FROM LOGIN SQLAUTHLOGIN`

Users that authenticate at the database

The following list shows possible syntax for users that can only be used in a contained database. The users created will not be related to any logins in the **master** database. The default schema and language options are not listed.

> **Security Note**
>
> This syntax grants users access to the database and also grants new access to the Database Engine.

- `CREATE USER [Domain1\WindowsUserBarry]`
- `CREATE USER [Domain1\WindowsGroupManagers]`
- `CREATE USER Barry WITH PASSWORD = 'sdjklalie8rew8337!$d'`

Users based on Windows principals without logins in master

The following list shows possible syntax for users that have access to the Database Engine through a Windows group but do not have a login in **master**. This syntax can be used in all types of databases. The default schema and language options are not listed.

This syntax is similar to users based on logins in master, but this category of user does not have a login in master. The user must have access to the Database Engine through a Windows group login.

This syntax is similar to contained database users based on Windows principals, but this category of user does not get new access to the Database Engine.

- `CREATE USER [Domain1\WindowsUserBarry]`
- `CREATE USER [Domain1\WindowsUserBarry] FOR LOGIN Domain1\WindowsUserBarry`

- `CREATE USER [Domain1\WindowsUserBarry] FROM LOGIN Domain1\WindowsUserBarry`
- `CREATE USER [Domain1\WindowsGroupManagers]`
- `CREATE USER [Domain1\WindowsGroupManagers] FOR LOGIN [Domain1\WindowsGroupManagers]`
- `CREATE USER [Domain1\WindowsGroupManagers] FROM LOGIN [Domain1\WindowsGroupManagers]`

Users that cannot authenticate

The following list shows possible syntax for users that cannot login to SQL Server.

- `CREATE USER RIGHTSHOLDER WITHOUT LOGIN`
- `CREATE USER CERTUSER FOR CERTIFICATE SpecialCert`
- `CREATE USER CERTUSER FROM CERTIFICATE SpecialCert`
- `CREATE USER KEYUSER FOR ASYMMETRIC KEY SecureKey`
- `CREATE USER KEYUSER FROM ASYMMETRIC KEY SecureKey`

Security

Creating a user grants access to a database but does not automatically grant any access to the objects in a database. After creating a user, common actions are to add users to database roles which have permission to access database objects, or grant object permissions to the user.

Special Considerations for Contained Databases

When connecting to a contained database, if the user does not have a login in the **master** database, the connection string must include the contained database name as the initial catalog. The initial catalog parameter is always required for a contained database user with password.

In a contained database, creating users helps separate the database from the instance of the Database Engine so that the database can easily be moved to another instance of SQL Server. For more information, see Contained Databases. To change a database user from a user based on a SQL Server authentication login to a contained database user with password, see sp_migrate_user_to_contained (Transact-SQL).

In a contained database, users do not have to have logins in the **master** database. Database Engine administrators should understand that access to a contained database can be granted at the database level, instead of the Database Engine level. For more information, see Security Best Practices with Contained Databases.

When using contained database users on Azure SQL Database, configure access using a database-level firewall rule, instead of a server-level firewall rule. For more information, see sp_set_database_firewall_rule (Azure SQL Database).

Permissions

Requires ALTER ANY USER permission on the database.

Examples

A. Creating a database user based on a SQL Server login

The following example first creates a SQL Server login named `AbolrousHazem`, and then creates a corresponding database user `AbolrousHazem` in `AdventureWorks2012`.

```
CREATE LOGIN AbolrousHazem
    WITH PASSWORD = '340$Uuxwp7Mcxo7Khy';
USE AdventureWorks2012;
GO
CREATE USER AbolrousHazem FOR LOGIN AbolrousHazem;
GO
```

B. Creating a database user with a default schema

The following example first creates a server login named `WanidaBenshoof` with a password, and then creates a corresponding database user `Wanida`, with the default schema `Marketing`.

```
CREATE LOGIN WanidaBenshoof
    WITH PASSWORD = '8fdKJl3$nlNv3049jsKK';
USE AdventureWorks2012;
CREATE USER Wanida FOR LOGIN WanidaBenshoof
    WITH DEFAULT_SCHEMA = Marketing;
GO
```

C. Creating a database user from a certificate

The following example creates a database user `JinghaoLiu` from certificate `CarnationProduction50`.

Applies to: SQL Server 2008 through SQL Server 2016.

```
USE AdventureWorks2012;
CREATE CERTIFICATE CarnationProduction50
    WITH SUBJECT = 'Carnation Production Facility Supervisors',
    EXPIRY_DATE = '11/11/2011';
GO
CREATE USER JinghaoLiu FOR CERTIFICATE CarnationProduction50;
GO
```

D. Creating and using a user without a login

The following example creates a database user `CustomApp` that does not map to a SQL Server login. The example then grants a user `adventure-works\tengiz0` permission to impersonate the `CustomApp` user.

```
USE AdventureWorks2012 ;
CREATE USER CustomApp WITHOUT LOGIN ;
GRANT IMPERSONATE ON USER::CustomApp TO [adventure-works\tengiz0] ;
GO
```

To use the `CustomApp` credentials, the user `adventure-works\tengiz0` executes the following statement.

```
EXECUTE AS USER = 'CustomApp' ;
GO
```

To revert back to the `adventure-works\tengiz0` credentials, the user executes the following statement.

```
REVERT ;
GO
```

E. Creating a contained database user with password

The following example creates a contained database user with password. This example can only be executed in a contained database.

Applies to: SQL Server 2012 through SQL Server 2016. This example works in SQL Database V12 if DEFAULT_LANGUAGE is removed.

```
USE AdventureWorks2012 ;
GO
CREATE USER Carlo
WITH PASSWORD='RN92piTCh%$!~3K9844 Bl*'
    , DEFAULT_LANGUAGE=[Brazilian]
    , DEFAULT_SCHEMA=[dbo]
GO
```

F. Creating a contained database user for a domain login

The following example creates a contained database user for a login named Fritz in a domain named Contoso. This example can only be executed in a contained database.

Applies to: SQL Server 2012 through SQL Server 2016.

```
USE AdventureWorks2012 ;
GO
CREATE USER [Contoso\Fritz] ;
GO
```

G. Creating a contained database user with a specific SID

The following example creates a SQL Server authenticated contained database user named CarmenW. This example can only be executed in a contained database.

Applies to: SQL Server 2012 through SQL Server 2016.

```
USE AdventureWorks2012 ;
GO
CREATE USER CarmenW WITH PASSWORD = 'a8ea v*(Rd##+'
, SID = 0x01050000000000090300000063FF0451A9E7664BA705B10E37DDC4B7;
```

See Also

- Create a Database User
- sys.database_principals (Transact-SQL)
- ALTER USER (Transact-SQL)
- DROP USER (Transact-SQL)
- CREATE LOGIN (Transact-SQL)
- EVENTDATA (Transact-SQL)
- Contained Databases

Community Additions