GRANT (Transact-SQL)

06/12/2017 • 6 minutes to read • Contributors

In this article

Syntax
Arguments
Remarks
WITH GRANT OPTION
Chart of SQL Server Permissions
Permissions
Examples
See Also

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

Grants permissions on a securable to a principal. The general concept is to GRANT <some permission> ON <some object> TO <some user, login, or group>. For a general discussion of permissions, see Permissions (Database Engine).

Transact-SQL Syntax Conventions

Syntax

Copy

```
-- Syntax for SQL Server and Azure SQL Database

-- Simplified syntax for GRANT
GRANT { ALL [ PRIVILEGES ] }
      | permission [ ( column [ ,...n ] ) ] [ ,...n ]
      [ ON [ class :: ] securable ] TO principal [ ,...n ]
      [ WITH GRANT OPTION ] [ AS principal ]
```

FEEDBACK

Copy

```
-- Syntax for Azure SQL Data Warehouse and Parallel Data Warehouse

GRANT
    <permission> [ ,...n ]
    [ ON [ <class_type> :: ] securable ]
    TO principal [ ,...n ]
    [ WITH GRANT OPTION ]
[;]

<permission> ::=
{ see the tables below }

<class_type> ::=
{
      LOGIN
    | DATABASE
    | OBJECT
    | ROLE
    | SCHEMA
    | USER
}
```

Arguments

ALL
This option is deprecated and maintained only for backward compatibility. It does not grant all possible permissions. Granting ALL is equivalent to granting the following permissions:

- If the securable is a database, ALL means BACKUP DATABASE, BACKUP LOG, CREATE DATABASE, CREATE DEFAULT, CREATE FUNCTION, CREATE PROCEDURE, CREATE RULE, CREATE TABLE, and CREATE VIEW.
- If the securable is a scalar function, ALL means EXECUTE and REFERENCES.
- If the securable is a table-valued function, ALL means DELETE, INSERT, REFERENCES, SELECT, and UPDATE.
- If the securable is a stored procedure, ALL means EXECUTE.
- If the securable is a table, ALL means DELETE, INSERT, REFERENCES, SELECT, and UPDATE.
- If the securable is a view, ALL means DELETE, INSERT, REFERENCES, SELECT, and UPDATE.

PRIVILEGES
Included for ISO compliance. Does not change the behavior of ALL.

permission
Is the name of a permission. The valid mappings of permissions to securables are described in the subtopics listed below.

column
Specifies the name of a column in a table on which permissions are being granted. The parentheses () are required.

class
Specifies the class of the securable on which the permission is being granted. The scope qualifier **::** is required.

securable
Specifies the securable on which the permission is being granted.

TO *principal*
Is the name of a principal. The principals to which permissions on a securable can be granted vary, depending on the securable. See the subtopics listed below for valid combinations.

GRANT OPTION
Indicates that the grantee will also be given the ability to grant the specified permission to other principals.

AS *principal*
Use the AS principal clause to indicate that the principal recorded as the grantor of the permission should be a principal other than the person executing the statement. For example, presume that user Mary is principal_id 12 and user Raul is principal 15. Mary executes `GRANT SELECT ON OBJECT::X TO Steven WITH GRANT OPTION AS Raul;` Now the sys.database_permissions table will indicate that the grantor_prinicpal_id was 15 (Raul) even though the statement was actually executed by user 13 (Mary).

Using the AS clause is typically not recommended unless you need to explicitly define the permission chain. For more information, see the **Summary of the Permission Check Algorithm** section of Permissions (Database Engine).

The use of AS in this statement does not imply the ability to impersonate another user.

Remarks

The full syntax of the GRANT statement is complex. The syntax diagram above was simplified to draw attention to its structure. Complete syntax for granting permissions on specific securables is described in the articles listed below.

The REVOKE statement can be used to remove granted permissions, and the DENY statement can be used to prevent a principal from gaining a specific permission through a GRANT.

Granting a permission removes DENY or REVOKE of that permission on the specified securable. If the same permission is denied at a higher scope that contains the securable, the DENY takes precedence. But revoking the granted permission at a higher scope does not take precedence.

Database-level permissions are granted within the scope of the specified database. If a user needs permissions to objects in another database, create the user account in the other database, or grant the user account access to the other database, as well as the current database.

> ⚠ **Caution**
>
> A table-level DENY does not take precedence over a column-level GRANT. This inconsistency in the permissions hierarchy has been preserved for the sake of backward compatibility. It will be removed in a future release.

The sp_helprotect system stored procedure reports permissions on a database-level securable.

WITH GRANT OPTION

The **GRANT** ... **WITH GRANT OPTION** specifies that the security principal receiving the permission is given the ability to grant the specified permission to other security accounts. When the principal that receives the permission is a role or a Windows group, the **AS** clause must be used when the object permission needs to be further granted to users who are not members of the group or role. Because only a user, rather than a group or role, can execute a **GRANT** statement, a specific member of the group or role must use the **AS** clause to explicitly invoke the role or group membership when granting the permission. The following example shows how the **WITH GRANT OPTION** is used when granted to a role or Windows group.

Copy

```
-- Execute the following as a database owner
GRANT EXECUTE ON TestProc TO TesterRole WITH GRANT OPTION;
EXEC sp_addrolemember TesterRole, User1;
-- Execute the following as User1
-- The following fails because User1 does not have the permission as the User1
GRANT EXECUTE ON TestMe TO User2;
-- The following succeeds because User1 invokes the TesterRole membership
GRANT EXECUTE ON TestMe TO User2 AS TesterRole;
```

Chart of SQL Server Permissions

For a poster sized chart of all Database Engine permissions in pdf format, see https://aka.ms/sql-permissions-poster.

Permissions

The grantor (or the principal specified with the AS option) must have either the permission itself with GRANT OPTION, or a higher permission that implies the permission being granted. If using the AS option, additional requirements apply. See the securable-specific article for details.

Object owners can grant permissions on the objects they own. Principals with CONTROL permission on a securable can grant permission on that securable.

Grantees of CONTROL SERVER permission, such as members of the sysadmin fixed server role, can grant any permission on any securable in the server. Grantees of CONTROL permission on a database, such as members of the db_owner fixed database role, can grant any permission on any securable in the database. Grantees of CONTROL permission on a schema can grant any permission on any object within the schema.

Examples

The following table lists the securables and the articles that describe the securable-specific syntax.

Application Role	GRANT Database Principal Permissions (Transact-SQL)
Assembly	GRANT Assembly Permissions (Transact-SQL)
Asymmetric Key	GRANT Asymmetric Key Permissions (Transact-SQL)
Availability Group	GRANT Availability Group Permissions (Transact-SQL)
Certificate	GRANT Certificate Permissions (Transact-SQL)
Contract	GRANT Service Broker Permissions (Transact-SQL)
Database	GRANT Database Permissions (Transact-SQL)
Database Scoped Credential	GRANT Database Scoped Credential (Transact-SQL)
Endpoint	GRANT Endpoint Permissions (Transact-SQL)
Full-Text Catalog	GRANT Full-Text Permissions (Transact-SQL)
Full-Text Stoplist	GRANT Full-Text Permissions (Transact-SQL)
Function	GRANT Object Permissions (Transact-SQL)
Login	GRANT Server Principal Permissions (Transact-SQL)

Message Type	GRANT Service Broker Permissions (Transact-SQL)
Object	GRANT Object Permissions (Transact-SQL)
Queue	GRANT Object Permissions (Transact-SQL)
Remote Service Binding	GRANT Service Broker Permissions (Transact-SQL)
Role	GRANT Database Principal Permissions (Transact-SQL)
Route	GRANT Service Broker Permissions (Transact-SQL)
Schema	GRANT Schema Permissions (Transact-SQL)
Search Property List	GRANT Search Property List Permissions (Transact-SQL)
Server	GRANT Server Permissions (Transact-SQL)
Service	GRANT Service Broker Permissions (Transact-SQL)
Stored Procedure	GRANT Object Permissions (Transact-SQL)
Symmetric Key	GRANT Symmetric Key Permissions (Transact-SQL)
Synonym	GRANT Object Permissions (Transact-SQL)
System Objects	GRANT System Object Permissions (Transact-SQL)
Table	GRANT Object Permissions (Transact-SQL)
Type	GRANT Type Permissions (Transact-SQL)
User	GRANT Database Principal Permissions (Transact-SQL)
View	GRANT Object Permissions (Transact-SQL)
XML Schema Collection	GRANT XML Schema Collection Permissions (Transact-SQL)

See Also

DENY (Transact-SQL)
REVOKE (Transact-SQL)

sp_addlogin (Transact-SQL)
sp_adduser (Transact-SQL)
sp_changedbowner (Transact-SQL)
sp_dropuser (Transact-SQL)
sp_helprotect (Transact-SQL)
sp_helpuser (Transact-SQL)

> **Note**
>
> The feedback system for this content will be changing soon. Old comments will not be carried over. If content within a comment thread is important to you, please save a copy. For more information on the upcoming change, we invite you to read our blog post.

0 Comments

Sign in

5 people listening

+ Follow

Post comment as...