



Principals (Part-2) : Database Users, Database Roles



Database Users

- After you've set up your server-level logins, you can create database users that map back to those logins, whether they're Windows or SQL Server logins.
- can also create database users that do not map to logins.
- These types of logins are generally used for contained databases, impersonation, or development and testing.
- SQL Server provides the CREATE USER statement for creating database users. You must run this statement within the context of the database in which the user is being defined.
 - USE Adventureworks;
CREATE USER [win10b\winuser01];
GRANT ALTER ON SCHEMA::Sales TO [win10b\winuser01];
- The winuser01 user is based on the win10b\winuser01 login.
- When you create a database user that has the same name as a login, you do not need to specify the login.



Database Users (contd..)

- If you want to create a user with a different name, you must include the FOR LOGIN or FROM LOGIN clause.
 - `CREATE USER winuser03 FOR LOGIN [win10b\winuser01];`
`GRANT ALTER ON SCHEMA::Sales TO winuser03;`
- You can create only one user in a database per login.
- If you want to try out both these statements, you'll need to drop the first user before creating the second.
- The two preceding examples also include a GRANT statement that assigns the ALTER permission to the user on the Sales schema. As a result, the user will be able to alter any object within that schema.
- When you grant a permission on a specific object, you must specify the type of object and its name, separated by the scope qualifier (double colons).
- In some GRANT statements, the securable is implied, so it does not need to be specified.



Database Users (contd..)

- Creating a database user that's associated with a SQL Server login is just as simple as creating a user based on a Windows login, especially when you use the same name
 - `CREATE USER sqluser01;`
- The `CREATE USER` statement creates the `sqluser01` user, but this time, the example grants no permissions.
- the user receives only the `CONNECT` permission, which you can verify by running the following `SELECT` statement:
 - `SELECT pe.state_desc, pe.permission_name FROM sys.database_principals pr INNER JOIN sys.database_permissions pe
ON pr.principal_id = pe.grantee_principal_id WHERE pr.principal_id = USER_ID('sqluser01');`



Database Roles

- A database role is a group of users that share a common set of database-level permissions.
- As with server roles, SQL Server supports both fixed and user-defined database roles.
- Database-level roles are database-wide in their permissions scope.
- Fixed-database roles are defined at the database level and exist in each database.
- There are also some special-purpose database roles in the msdb database.
- Members of the **db_owner** database role can manage fixed-database role membership.
- You can add any database account and other SQL Server roles into database-level roles.
- Server-level permissions cannot be granted to database roles.
- Logins and other server-level principals (such as server roles) cannot be added to database roles.

Fixed Database Roles

Fixed-Database role name	Description
db_owner	Members of the db_owner fixed database role can perform all configuration and maintenance activities on the database, and can also drop the database in SQL Server.
db_securityadmin	Members of the db_securityadmin fixed database role can modify role membership for custom roles only and manage permissions. Members of this role can potentially elevate their privileges and their actions should be monitored.
db_accessadmin	Members of the db_accessadmin fixed database role can add or remove access to the database for Windows logins, Windows groups, and SQL Server logins.
db_backupoperator	Members of the db_backupoperator fixed database role can back up the database.
db_ddladmin	Members of the db_ddladmin fixed database role can run any Data Definition Language (DDL) command in a database.
db_datawriter	Members of the db_datawriter fixed database role can add, delete, or change data in all user tables.
db_datareader	Members of the db_datareader fixed database role can read all data from all user tables and views. User objects can exist in any schema except sys and INFORMATION_SCHEMA.
db_denydatawriter	Members of the db_denydatawriter fixed database role cannot add, modify, or delete any data in the user tables within a database.
db_denydatareader	Members of the db_denydatareader fixed database role cannot read any data from the user tables and views within a database.

Database Level Roles and Permissions

CONTROL DATABASE: Has all permissions in the database

db_datareader

GRANT SELECT ON DATABASE::

db_denydatareader

DENY SELECT ON DATABASE::

db_datawriter

GRANT INSERT ON DATABASE::

GRANT UPDATE ON DATABASE::

GRANT DELETE ON DATABASE::

db_denydatawriter

DENY INSERT ON DATABASE::

DENY UPDATE ON DATABASE::

DENY DELETE ON DATABASE::

db_accessadmin

CREATE SCHEMA

ALTER ANY USER

CONNECT

db_securityadmin

ALTER ANY ROLE, CREATE ROLE

ALTER ANY APPLICATION ROLE

VIEW DEFINITION

public

There are no database-level permissions inherent in the public database role, however some database permissions are present by default. Specifically, VIEW ANY COLUMN MASTER KEY DEFINITION, VIEW ANY COLUMN ENCRYPTION KEY DEFINITION, and SELECT permission on many individual system tables. These permissions can be revoked.

db_backupoperator

BACKUP DATABASE

BACKUP LOG

CHECKPOINT

db_ddladmin

ALTER ANY ASSEMBLY
ALTER ANY ASYMMETRIC KEY
ALTER ANY CERTIFICATE
ALTER ANY CONTRACT
ALTER ANY DATABASE DDL TRIGGER
ALTER ANY DATABASE EVENT NOTIFICATION
ALTER ANY DATASPACE
ALTER ANY FULLTEXT CATALOG
ALTER ANY MESSAGE TYPE
ALTER ANY REMOTE SERVICE BINDING
ALTER ANY ROUTE
ALTER ANY SCHEMA
ALTER ANY SERVICE
ALTER ANY SYMMETRIC KEY
CHECKPOINT
CREATE AGGREGATE
CREATE DEFAULT
CREATE FUNCTION
CREATE PROCEDURE
CREATE QUEUE
CREATE RULE
CREATE SYNONYM
CREATE TABLE
CREATE TYPE
CREATE VIEW
CREATE XML SCHEMA COLLECTION
REFERENCES

There are various special purpose roles in the msdb database

The other database level permissions, are not granted to any fixed database role except db_owner.

SQL Server 2017



User defined Database Roles

- To set up a user-defined database role, you must create the role, grant permissions to the role, and add members to the role (or add members and then grant permissions).
 - `CREATE ROLE dbdev;`
`GRANT SELECT ON DATABASE::WideWorldImporters TO dbdev;`
`ALTER ROLE dbdev ADD MEMBER [win10b\winuser01];`
`ALTER ROLE dbdev ADD MEMBER sqluser01;`