**Roles**

In SQL Server, roles are a powerful tool for managing permissions by grouping users with similar access requirements together. Roles simplify permission management by allowing you to grant permissions to a role and then add or remove users from that role as needed, rather than managing permissions for each user individually.

In SQL Server, there are two main types of roles: server roles and database roles. Each type serves a distinct purpose and operates at a different level of the SQL Server architecture.

1. Server Roles:

Server-level roles control access to server-wide resources and operations. These roles are managed at the server level and apply to the entire SQL Server instance.

Types of Server Roles:

* Fixed Server Roles: Fixed server roles in SQL Server are predefined roles that come with specific permissions and are designed to control access to server-wide resources and operations. These roles provide a convenient way to grant common administrative privileges without having to manually assign individual permissions.

There are several fixed server roles, each with its own set of permissions. Here's an overview of some commonly used fixed server roles:

1. sysadmin: Members of the sysadmin fixed server role have unrestricted access to the entire SQL Server instance. They can perform any action on the server, including configuration changes, database creation, and security management. This role is typically reserved for system administrators.
2. serveradmin: Members of the serveradmin fixed server role can change server-wide configuration options and shut down the server. They cannot create or modify logins or databases.
3. setupadmin: Members of the setupadmin fixed server role can add or remove linked servers and perform other setup-related tasks.
4. securityadmin: Members of the securityadmin fixed server role can manage server-level security, including adding and removing logins and assigning permissions.
5. processadmin: Members of the processadmin fixed server role can end processes running on the SQL Server instance.
6. bulkadmin: Members of the bulkadmin fixed server role can run the bulk insert command and load data into a database.
7. diskadmin: Members of the diskadmin fixed server role can manage disk files.
8. public: Every login belongs to the public server role by default that means permissions granted to the public role are inherited by all logins.

Example: Adding a Login to a Fixed Server Role

Let's say you want to add a login named MyLogin to the sysadmin fixed server role:

ALTER SERVER ROLE diskadmin ADD MEMBER MyLogin;

By adding MyLogin to the diskadmin role, you're granting it managing disk files privileges.

Note: you need to have permissions to add server principal.

* Custom Server Roles: These are roles that you can create to define custom sets of permissions tailored to your specific requirements.

You can create custom server roles using the CREATE SERVER ROLE statement:

CREATE SERVER ROLE RoleName;

Once you've created a custom server role, you can grant permissions to the role using the GRANT statement, similar to how you grant permissions to users within a database.

Example:

Let's create a custom server role named BackupAdmin:

-- Create a Custom Server Role

CREATE SERVER ROLE BackupAdmin;

Next, grant the necessary permissions to the BackupAdmin role:

-- Assign a permission to the role; for example, the ALTER ANY LOGIN permission grants the ability to create new logins and drop existing.

Use Master;

GRANT ALTER ANY LOGIN TO BackupAdmin;

-- Assign server role to a server user

ALTER SERVER ROLE BackupAdmin ADD MEMBER MyLogin

Managing Custom Server Roles:

You can manage custom server roles by adding or removing members, altering permissions, and dropping roles as needed. Use the ALTER SERVER ROLE and DROP SERVER ROLE statements to make changes to custom roles.

ALTER SERVER ROLE RoleName ADD MEMBER LoginName;

ALTER SERVER ROLE RoleName DROP MEMBER LoginName;

DROP SERVER ROLE RoleName;

Example: ALTER SERVER ROLE RoleName ADD MEMBER MyLogin;

ALTER SERVER ROLE RoleName DROP MEMBER MyLogin;

DROP SERVER ROLE BackupAdmin;

2. Database Roles:

Database level roles control access to specific databases and their objects. These roles are managed within each database and apply only to objects within that database.

Types of Database Roles:

* Fixed Database Roles: Similar to fixed server roles, these are predefined roles that come with each database, each with a specific set of permissions.

|  |  |
| --- | --- |
| **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. (In SQL Database and Azure Synapse, some maintenance activities require server-level permissions and cannot be performed by db\_owners.) |
| 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. |
| 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 in the user tables within a database. |

Example: Let's say you want MyUser to have permissions as db\_datareader:

use test;

ALTER ROLE db\_datareader ADD MEMBER MyUser;

* Custom Database Roles: These are roles that you can create within a database to define custom sets of permissions for specific groups of users.

Example: Let's create a custom database role named SalesRole in the AdventureWorks database and grant DELETE permission on the city table to that role:

USE test;

CREATE ROLE SalesRole;

GRANT DELETE ON city TO SalesRole;

Now, let's add a user to the SalesRole:

ALTER ROLE SalesRole ADD MEMBER MyUser;

And then, from MyUser side:

use test

delete from city

and of course: insert into city values(1,'fsdfsd'); is not permitted.