Ownership and User-Schema Separation in SQL Server

03/30/20172 minutes to readContributors all

In this article

A core concept of SQL Server security is that owners of objects have irrevocable permissions to administer them. You cannot remove privileges from an object owner, and you cannot drop users from a database if they own objects in it.

User-Schema Separation

User-schema separation allows for more flexibility in managing database object permissions. A *schema* is a named container for database objects, which allows you to group objects into separate namespaces. For example, the AdventureWorks sample database contains schemas for Production, Sales, and HumanResources.

The four-part naming syntax for referring to objects specifies the schema name.

Copy

```
Server.Database.DatabaseSchema.DatabaseObject
```

Schema Owners and Permissions

Schemas can be owned by any database principal, and a single principal can own multiple schemas. You can apply security rules to a schema, which are inherited by all objects in the schema. Once you set up access permissions for a schema, those permissions are automatically applied as new objects are added to the schema. Users can be assigned a default schema, and multiple database users can share the same schema.

By default, when developers create objects in a schema, the objects are owned by the security principal that owns the schema, not the developer. Object ownership can be transferred with ALTER AUTHORIZATION Transact-SQL statement. A schema can also contain objects that are owned by different users and have more granular permissions than those assigned to the schema, although this is not recommended because it adds complexity to managing permissions. Objects

can be moved between schemas, and schema ownership can be transferred between principals. Database users can be dropped without affecting schemas.

Built-In Schemas

SQL Server ships with ten pre-defined schemas that have the same names as the built-in database users and roles. These exist mainly for backward compatibility. You can drop the schemas that have the same names as the fixed database roles if you do not need them. You cannot drop the following schemas:

- `dbo`
- `guest`
- `sys`
- `INFORMATION_SCHEMA`

If you drop them from the model database, they will not appear in new databases.

> **Note**
>
> The `sys` and `INFORMATION_SCHEMA` schemas are reserved for system objects. You cannot create objects in these schemas and you cannot drop them.

The dbo Schema

The `dbo` schema is the default schema for a newly created database. The `dbo` schema is owned by the `dbo` user account. By default, users created with the CREATE USER Transact-SQL command have `dbo` as their default schema.

Users who are assigned the `dbo` schema do not inherit the permissions of the `dbo` user account. No permissions are inherited from a schema by users; schema permissions are inherited by the database objects contained in the schema.

> **Note**
>
> When database objects are referenced by using a one-part name, SQL Server first looks in the user's default schema. If the object is not found there, SQL Server looks next in the `dbo` schema. If the object is not in the `dbo` schema, an error is returned.

External Resources

For more information on object ownership and schemas, see the following resources.

Resource	Description
User-Schema Separation in SQL Server Books Online	Describes the changes introduced by user-schema separation. Includes new behavior, its impact on ownership, catalog views, and permissions.

See Also

Securing ADO.NET Applications
Application Security Scenarios in SQL Server
Authentication in SQL Server
Server and Database Roles in SQL Server
Authorization and Permissions in SQL Server
ADO.NET Managed Providers and DataSet Developer Center