# Sql server change table schema to dbo

This article provides an overview of sql schema and its use in SQL Server. Introduction to the SQL Schema database contains several objects, such as tables, views, stored processes, functions, indexes, triggers. An SQL schema is defined as a logical database of database objects. The owner of the scheme owns the scheme. This is a useful mechanism for warming database objects for various applications, access rights, managing security administration databases. There are no restrictions on the number of objects in the scheme. 2005 SQL Server, we have different user meanings and schematics. The owner of the database objects is a schema and we define the owners of the schema. We can have one or more owners of the scheme. Provides the following advantages: Ownership of sql schema can be quickly transferred to another user The scheme can be shared between multiple users Allows you to move database objects between database objects over database objects Access and security We define the default SQL scheme for the database user in the database creation user window: If we do not specify any default schema for the user, SQL Server provides dbo as the default schema. The default schema can be verified for the user using the following system function: . Get all the scheches and their owners in the database You can query sys.schemas system table to find schema in the database and their owners: SELECT s.name AS schema_name, INNER JOIN sys.sysusers u ON u.uid = s.principal_id On the good screenshot, we see the schema of the main database and their owners: SQL Scheme is a database entity, but you would u u uid the database for you to get a second result. For example, the AdventureWorks database displays the following schema information: Note: SQL Server does not allow users to create objects in sys and INFORMATION_SCHEMA. Used to store objects of the internal system. Let us create a new table in the AdventureWorks database with the following query: We did not specify any schema in the CREATE TABLE statement. Automatically applies the dbo schema to the table because the current default schema is user dbo: Specify a default SQL SCHEMA while creating a new login We create a new SQL login and a database user with a default schema other than dbo. Go to Security and create a new login: In general, specify the following entries: Enter the SQL user name Select sql Server authentication method and enter the password for the user Locate the User Mapping page and check the AdventureWorks database: Click on the eclipse for the default schema and browse the schema available in the database. Select the [Person] scheme as the default schema for this user: Click OK and return to the user mapping page. We can see that the new login has access to the AdventureWorks database and its default schema is [Person]: Click OK to create an SQL login, user of the database in the AdventureWorks database. Connect to SQL instances with this login. create table without specifying the SQL Schema. We can see that it creates a table in the default schema for the user, that is, [Person]: Alternatively, we can run the script by specifying the schema name. Script to create a table in the [Person] schema: CREATE A TABLE [person].[DemoSchema] Script to create a table in the dbo schema: CREATE A TABLE [dbo]. [DemoSchema] Create a new SQL scheme on SQL Server We can create schemas to our requirements. We can create a schema in a specific database. Let's create a new schema in the AdventureWorks database. Expand the AdventureWorks database and go to Security: right-click the schemas and click on New Schema: Specify a new schema name and schema owner. You can search for the owner of the schema from existing logins: Instead of creating a schema, click on create scripts, to give you the equivalent of a T-SQL statement: CREATE SCHEMA [Customer] AUTHORIZATION [Demouser] In this syntax, We see that: [Customer] is the new schema name that we want to create u database AdventureWorks We are tired of the owner of the scheme using the keyword AUTHORIZATION in the command CREATE SCHEMA Execute coman and refresh the list of schematics (right-clicking on the schema i refresh). In this list, you can view the new [Customer]: Note: We previously created a demo user using the default [Person] scheme. The demo user also owns the customer scheme scheme. If we connect to SQL instances using a user demo and create objects without specifying a schema name, it still requires a default schema as [Person]. Changing the SQL schema of an existing object on SQL Server Can transfer objects to different schemas as well. For example, we mms the table from [dbo] owner to sales scheme in the AdventureWorks database by following these steps: Right-click on the specified table name and select Design Option: Opens table designer. With this designer, we can change table properties, such as column data types, default values, schema. Click On Properties as shown in the following illustration: Opens table properties. Displays table schema, server name, identity column, lock escalation, and file group information: Click on SQL SCHEMA and open the schema that is available in the database: Select the required schema [Sales], and gives a warning: Changing the schema of this object will cause all current permissions for this object to fall Click on Yes to continue: Close the table designer and save the changes after clicking Yes : Refresh the tables in the database and we can see that the schema of this table changes from dbo to sale: SQL Server searches for the object in the default schema and dbo schema by default. If the object belongs to the other than the default and dbo schema, we must specify the schema name during access to the object. If the object does not exist in the default schema or dbo schema, you receive an error message: We can specify a schema name in this format to access the object. In this command is the [sale] scheme and [Demoschema] is the table: Decision A the scheme is a useful concept of the database. Helps us create logical grouping of objects, such as tables, stored processes, and functions. I just posted this on a similar question: In sql server 2005, how to change the schema table without losing data? A slight improvement in SAeid's excellent response... I added an exemo to the code to self-exhip, I added to the uniju at the top how i could change the sheath of both tables I saved procedure: DECLARE CURSORE CURSORE FOR SELECT SPECIFIC_SCHEMA as 'schema', specific_name AS 'name' INFORMATION_SCHEMA.routines WHERE specific_schema &lt;&gt; 'dbo' UNION ALL SELECT TABLE_SCHEMA AS 'schema', TABLE_NAME AS 'name' FROM INFORMATION_SCHEMA. TABLES where TABLE_SCHEMA &lt;&gt; dbo declares @schema sysname, @tab sysname, @sql varchar(500) OPEN CURSORE FETCH NEXT FROM CURSORE INTO @schema, @tab WHILE @@FETCH_STATUS = 0 BEGIN SET @sql = ALTER SCHEMA dbo TRANSFER [ + @schema + ]. [' + @tab +']' PRINT @sql exec (@sql) FETCH

NEXT FROM CURSORE INTO @SCHEMA, @tab END CLOSE CURSORE DEALLOCATE CURSORE I had to restore a dbdump, and found that the scheme wasn't dbo - I spent hours trying to get Sql Server management studio or visual studio data transfers to alter destination the scheme... In the end I'm just running this against a renewed smut on the new server to get things as I wanted. Today I encountered a problem where all my SQL Server tables used a different schema than dbo and the application could not understand different schemas. For example, if you use the ALTER SCHEMA statement, you can change the table schema from kompmsavator to dbo. ALTER SCHEMA dbo TRANSFER compmsauser.tablename This works perfectly unless you need to change hundreds of table scheas. The following query will generate the T-SQL needed to change each table: (change the WHERE clause to the schema you need to replace) SELECT ALTER SCHEMA dbo TRANSFER + s.Name +'. + o.Name FROM sys. Objects about INNER JOIN sys. Schemas on o.schema_id = s.schema_id WHERE s.Name = compmsauser In (o.Type = 'U' Or o.Type = 'P' Or o.Type = 'V') This query will create an ALTER SCHEMA statement for you! All you have to do now is copy and paste all the results in a new query window and execute. Please sign up or register to answer this question. Sign in or register to add a comment. Summary: In this tutorial you will learn how to use SQL Server ALTER SCHEMA to transfer the security from one schema to another. OVERVIEW SQL SERVER ALTER SCHEMAIASCHEMA shows that the security is transferring from a schema to another within the same database. Note that the security paper is the resource to which the database authorization system controls access. For example, a table is a security. This shows the syntax of the ALTER SCHEMA:ALTER SCHEMA target_schema_name TRANSFER statement [ entity_type ::: ] securable_name; This syntax:target_schema_name the schema name the current database to which you want to move the object. Note that it cannot be SYS or INFORMATION_SCHEMA. In entity_type, you can view an object, type, or collection of XML schemas. By default, it is in the object. Entity_type the class of entity for which the owner changed.object_name is the name of the security you want to transfer to the target_schema_name. If you move a stored process, function, view, or trigger, SQL Server will not change the schema name of these schemas. Therefore, it is recommended to drop and recreate these objects in the new schema instead of using the ALTER SCHEMA statement to move around. If you move an object, such as a <a0></a0> , you can use the You must manually change the references to reflect the new schema name. For example, if you move a table that references in a saved process, you must change the saved process to reflect the new schema name. SQL Server ALTER SCHEMA exampleA, create a new named office table in the dbo:CREATE TABLE dbo.offices (office_id INT PRIMARY KEY IDENTITY, office_name NVARCHAR(40) NOT NULL, office_address NVARCHAR(255) NI NULL, VARCHAR(20) phone); Then insert a few rows into the dob.offices table:INSERT INTO dbo.offices(office_name, office_address) VALUES ('Silicon Valley','400 North 1st Street, San Jose, CA 95130'), ('Sacramento','1070 River Dr., Sacramento, CA 95820'); Then create a saved process that looks for office id:CREATE PROC usp_get_office_by_id(@id INT ) AS BEGIN SELECT * FROM dbo.offices WHERE office_id = @id; END; After that, download this dbo.offices table in the sales scheme:ALTER SCHEMA sales TRANSFER OBJECT::d bo.offices; If you perform a usp_get_office_by_id operation, SQL Server will issue an error:Msg 208, Level 16, Status 1, Procedure usp_get_office_by_id, Row 5 [Batch Start Line 30] Invalid object name dbo.offices. Finally, manually change the saved process to reflect the new scheme:ALTER PROC usp_get_office_by_id( @id INT ) AS THE BEGINNING SELECT * FROM THE SALES.OFFICES WHERE OFFICE_ID = @id; END; In this tutorial, you learned how to use the SQL Server ALTER SCHEMA statement to transfer a security from one schema to another within the same database. Database.