Using SSMS to Authorize ownership when restoring Databases

Tips and Hints:

We can Backup and Restore SQL Server Databases – see SQL Server TipsnHints – Backup and Restore Databases.

However sometimes you will receive the following error as you are not authorised or have permissions to access the Database.

Graphical user interface, text, application, email

Description automatically generated

You will need to assign authorisation for your SQL Server account to access the Database(s) as follows:

|  |
| --- |
| -- Microsoft SQL Server, Error:15174 when restoring .bak SQL DBs.  -- Login 'DomainName\abc'-- owns one or more database(s).  -- How to change the owner of da-- Change the owner of database(s)before dropping the login.  --To check the ownership of database "AdventureWorks2019"  SELECT name as [DB Name],  suser\_sname(owner\_sid) as [Owner]  FROM sys.databases  WHERE name='AdventureWorks2019'  --Change the ownership of database "AdventureWorks2019" to sa  ALTER AUTHORIZATION ON DATABASE::AdventureWorks2019 TO sa;  --Change the ownership of database "TSQL" to sa  ALTER AUTHORIZATION ON DATABASE::TSQL TO sa; |

Simply modify the below author authorisation by changing TSQL to the restored database name.

For Reference Only

How to amend authorisation on Database, Schemas or Tables

We create, modify and drop databases, objects and other SQL Server related entities like logins, users, schemas, roles, endpoints, etc. as per our needs to address database objectives. Sometimes, some of the database activities stop working and start throwing errors because of having the wrong principals as their owner. In this article, I will explain how to fix issues that are related to incorrect or wrong ownership assigned to various securables.

Solution

Every securable in SQL Server has an owner. SQL Server assigns the owner to the securable during creation to secure and ensure permanent access to those objects by owners so the object can be managed. The idea behind assigning ownership as part of SQL Server security is that the owner of any securable must have unchangeable permissions and prevent any kind of lockdown situation in case access is revoked for that securable. We cannot remove privileges from an object owner and neither we can drop any principal (users or logins) if they own objects.

If you need to drop users or logins from SQL Server, then you need to change the ownership first. Database and object ownership can be transferred using the ALTER AUTHORIZATION command. You either need to be part of the sysadmin fixed server role or have at least TAKE OWNERSHIP permission on the database and IMPERSONATE permission on the new owner login to run ALTER AUTHORIZATION.

**Note:** Ownership of some entities in SQL Server cannot be changed like linked servers, statistics, constraints, rules, defaults, triggers, Service Broker queues, credentials, partition functions, partition schemes, database master keys, service master key and event notifications.

When ownership is transferred, permissions on schema-contained objects that do not have explicit owners will be dropped. We cannot change the owner of sys, dbo, or information\_schema.

Below, I will show how to assign or transfer ownership of various entities in SQL Server.

## Change Database Ownership in SQL Server

Databases are owned by the login used when the database was created. We can change database ownership to any server level principal or login after database creation.

Let’s understand using an example. You have created a database using your login and did not change the owner to some login like sa. Later if you leave the organization and somebody from the DBA team tries to drop your login to clean up the system, SQL Server will not allow the login to be dropped because the login owns a database. If you try this, you will get the below error.

Login 'DomainName\abc' owns one or more database(s).  
Change the owner of database(s) before dropping the login. (Microsoft SQL Server, Error:15174)

The solution to fix this issue is to change the ownership of the database to some other login and then drop the login you no longer need. Let’s demonstrate it in step-by-step process on how to change the database ownership.

The first step is to check database ownership of identified database by running the below command.

--Check the ownership of database "AdventureWorks2019"

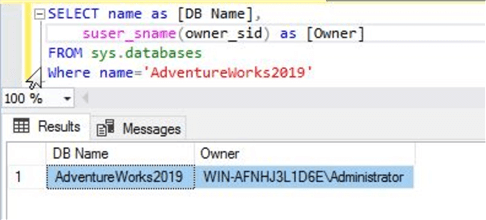
SELECT name as [DB Name],

suser\_sname(owner\_sid) as [Owner]

FROM sys.databases

WHERE name='AdventureWorks2019'

You can see the database is owned by a local administrator account.

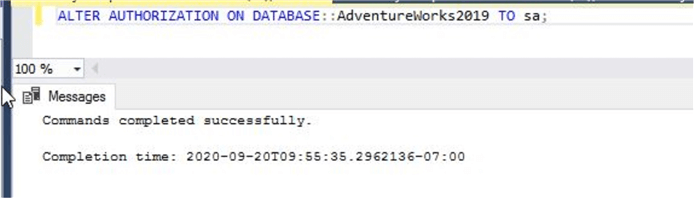


Now, let’s change the ownership to the sa account by running the below ALTER AUTHORIZATION statement.

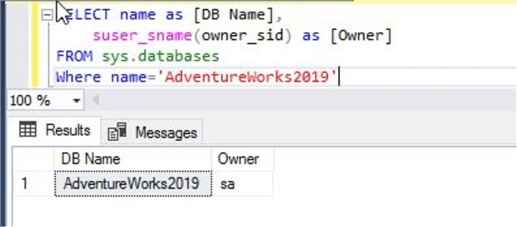
--Change the ownership of database "AdventureWorks2019" to sa

ALTER AUTHORIZATION ON DATABASE::AdventureWorks2019 TO sa;

Have a look at below screenshot of above command.



Now you can run the first command again to verify the ownership and you can see the ownership now shows as sa. Now you if you wanted you could drop the login that was throwing error 15174.



## Change Table Ownership in SQL Server

Let’s discuss how to change the ownership of tables. There are many requirements under which we need to change the ownership of database objects like tables, views or procedures. ALTER AUTHORIZATION can be used to change the ownership of all such database objects.

First run the below T-SQL statements to get the table owner for table "Address" in the AdventureWorks2019 database.

--Check the ownership of table "Address"

SELECT name as [Table Name],

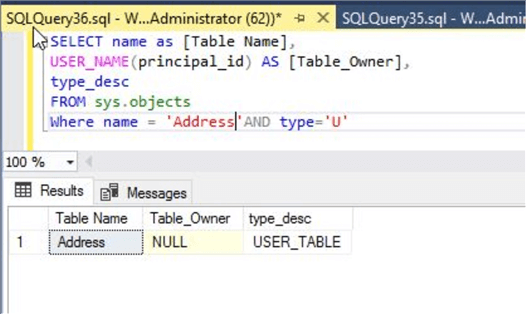
USER\_NAME(principal\_id) AS [Table\_Owner],

type\_desc

FROM sys.objects

WHERE name = 'Address' AND type='U'

You can see the table owner is NULL in the below screenshot.



Now let’s go ahead and assign an owner to this table. Run the below ALTER AUTHORIZATION statement to change the table owner to user "test". I had created the test login and user just for demonstration purpose, so you can use any login and user in your database.

--Change the ownership of table "Address"

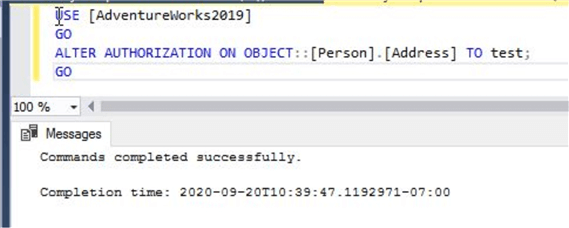
USE [AdventureWorks2019]

GO

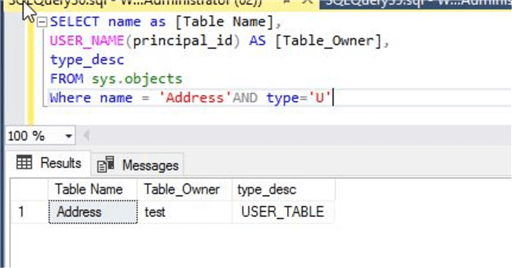
ALTER AUTHORIZATION ON OBJECT::[Person].[Address] TO test;

GO

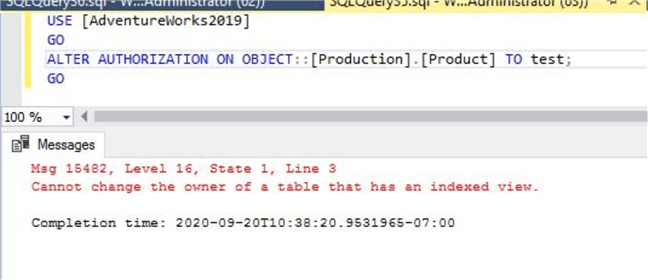
You can see the command completed successfully in the below image.



Below we can see the ownership has changed.



If any of your tables have indexed views, then you cannot change ownership and you will get the below error if you try to change ownership. You can see I am changing the table owner of table "Product" to user test in the below screenshot, but it gives me an error that the table has an indexed view so we cannot change the owner of the table.  We would need to drop the indexed view first, make the change and then recreate the indexed view.



## Change Schema Ownership in SQL Server

This section will educate you on two points. One is about how to check a schema owner and another will teach you how to change the existing owner of a schema to another owner.

Run the below command to check the schema owner. You can see I am checking ownership of schema "Person".

--Check the ownership of schema "Person"

USE AdventureWorks2019

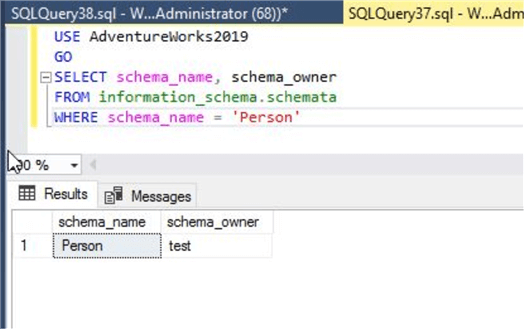
GO

SELECT schema\_name, schema\_owner

FROM information\_schema.schemata

WHERE schema\_name = 'Person'

The owner of schema "Person" is showing as test in the below screenshot. Now we need to change it to dbo or some other user which you want to assign.



Run the below ALTER AUTHORIZATION statement to change the owner of schema "Person" from test to dbo.

--Change the ownership of schema "Person"

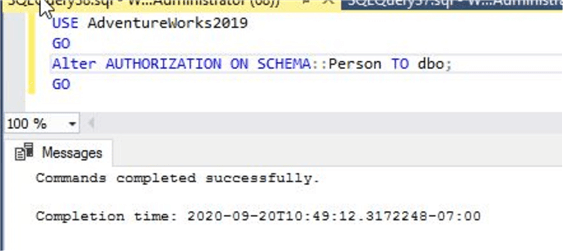
USE AdventureWorks2019

GO

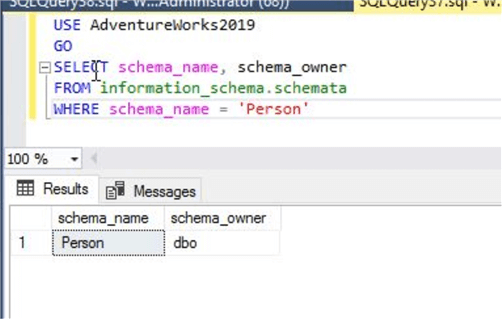
Alter AUTHORIZATION ON SCHEMA::Person TO dbo;

GO

The command executed successfully.



Now, run the above command again to see the schema owner change.



## Change Endpoint Ownership in SQL Server

Endpoints are server level securables which needs a server level login as their owner. You cannot assign a database level principal like users as the owner for an endpoint. If you are running Always On Availability Groups and your endpoint being used for AOAG has the wrong or incorrect owner, then your AOAG might have a problem. Here, I will show you how to check the ownership of endpoints and how to change endpoint owners.

Run the below command to get the owner of all the endpoints on your SQL Server instance.

--Check the ownership of Endpoint "TSQL Default TCP"

USE master

GO

select name as [EndPoint Name],

USER\_NAME(principal\_id) AS [EndPoint\_Owner],

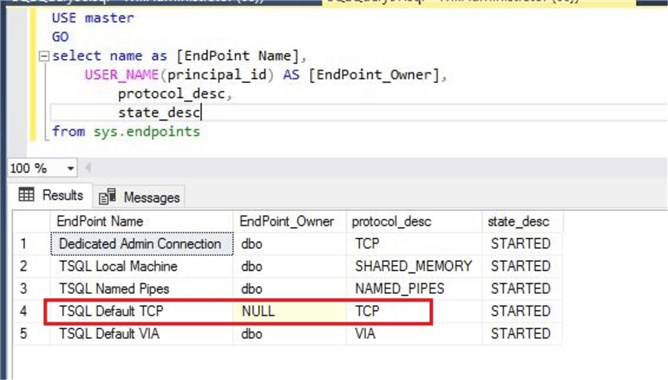
protocol\_desc,

state\_desc

from sys.endpoints

GO

You can see one endpoint has a NULL value as its owner. We need to change it to a correct login.



I ran the below command to change the ownership of this endpoint "TSQL Default TCP". You can see I used ALTER AUTHORIZATION to change its ownership to sa.

--Change the ownership of Endpoint "TSQL Default TCP"

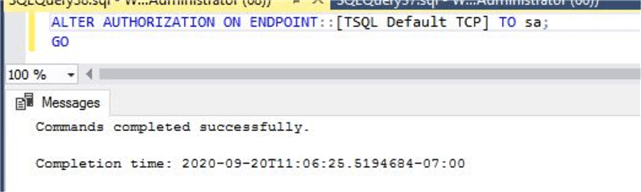
USE master

GO

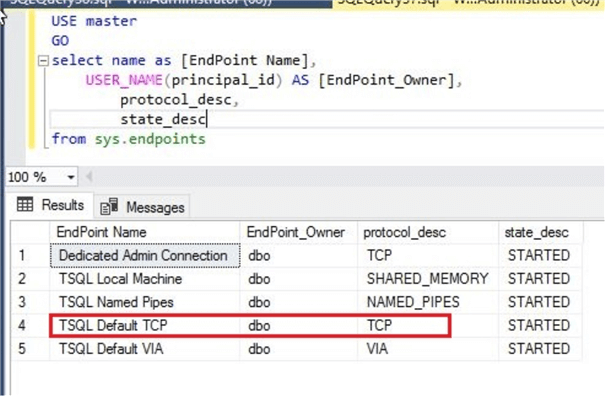
ALTER AUTHORIZATION ON ENDPOINT::[TSQL Default TCP] TO sa;

GO

The command executed successfully as shown below.



Now, we will again validate the ownership of all endpoints by running the same command that we ran in the first step of this section. You can now see the ownership of the endpoint shows as dbo.



Although this example is to set the correct ownership of an endpoint, whatever login is used for the endpoint owner cannot be dropped until the endpoint ownership is transferred to another login. This can be very useful in case you are stuck in such a scenario and you are struggling to transfer ownership. If any login owns an endpoint and you are trying to drop that login then you will get the below error:

Drop failed for Login "DomainName\abc"  
The server principal owns one or more endpoint(s) and cannot be dropped. (Microsoft SQL Server, Error:15141)

Use ALTER AUTHORIZATION to change the endpoint ownership before dropping the login.