Database (Sql Server)

# Security

The security aspects of the sql server

## Principals

* Think of it as a user group
* Maintained at two levels

1. Server level
2. Database level

### Useful queries

Server Level

1. SELECT \* FROM sys.server\_principals;

Select server principals. Contains all the principals for the instance of sql server

1. CREATE LOGIN DoNotDoThis WITH PASSWORD ='password'

Create sql login “DoNotDoThis” with password

1. CREATE LOGIN [<Domain>\nobody] FROM WINDOWS

Create login for windows user. The user is “nobody” and must be created before login

Database Level

1. SELECT \* FROM [security principals].sys.database\_principals

Select the principals for the database only. In this case the database [security principals] is created by using command “Create database [security principals]”

* Each database has it’s own set of principles

1. All the new database is created from the “model” system database.
2. CREATE USER [SWAGLAP\nobody]

Create user on database level

1. ALTER USER [swaglap\nobody] WITH NAME=noone

Change the user “swaglap\nobody” with name “noone”

1. Create user whoami without login

Create user with name “whoami” without any login details. This user will be present in sys.database\_principals but not in sys.server\_principals

1. DROP USER whoami

Delete the user whoami from database scope

## Objects

1. All the things (i.e. table, index, stored procedure etc) are stored as objects in sql server
2. That’s why there is a “object explorer” in SSMS

## Authorization

1. ALTER LOGIN [SWAGLAP\nobody] DISABLE

Disable the login for windows user “nobody” with domain “swaglap”. Th user nobody will not be login into the database server

1. ALTER LOGIN [SWAGLAP\nobody] ENABLE

To enable the user

## Permissions

1. SELECT \* FROM sys.fn\_my\_permissions(null,'server');

Check the permissions that the currently logged in user has in server. Put “database” in place of “server” to figure out the permissions on database.

1. Currently “nobody” can only view database and log on to the server
2. GRANT CONTROL SERVER TO [swaglap\nobody]

Grant “control server” privileges to nobody now

1. SELECT \* FROM sys.fn\_my\_permissions('numbers','object')

Table name: number

This will check the permission that nobody has on table “number”

1. Revoke takes precedence over grant
2. GRANT SELECT ON saucer.planets TO gorty

Grant select permission to gorty on saucer.planets table

## Impersonation

1. Use the “setuser” command to impersonate (i.e. to act on behalf of) user
2. User only “setuser” to get back to the original state
3. Use the “EXECUTE AS USER ='gorty'” because setuser command id deprecated. Use the “revert” command to get back to the original state.

## Group and Roles

1. Windows group / database role is a principal

### Server roles

1. Server roles are fixed

sp\_helpsrvrole -- to see all the server roles

sp\_srvrolepermission 'sysadmin' -- all the permissions that sysadmin has

## Schema

1. 3rd name in the 4 part name convention

<server name>.<database name>.<**schema name**>.<object name>

1. SELECT \* FROM securityII.sys.schemas

Select the schemas available in database “securityII”

Each schema has an owner and listed in sys.schema table

1. “dbo” user is the admin
2. Schema names within the same database needs to be unique