SECURITY

The Principle of Least Privilege (PLP)

- Users should be granted only the minimum permissions needed to perform their jobs.
- Restrict the number of sys admins
- Stay up to date on latest security trends

There are many levels of security in SQL Server

- Physical machine and Operating system
- Who has access and at what times - Server should be in a locked room with access limited to systems admins only
- Access the database instance via management studio rather than remotely accessing the physical machine
- Keep servers patched for latest security holes but test software updates extensively before rolling them into produc tion
- Don’t install unnecessary software or feature
- Disable unnecessary services and feature

- Instance-level
  - Defines who can login to the SQL Server instance and what permissions they have
  - sa - has instance-wide administration privileges
  - Anyone who can login to the instance is called a ’login’
- Database-level
  - Defines which logins can use which databases
  - Anyone who can login to a database is called a ’User’

Login vs User

- Login – access to the SQL Server instance
- User – access to the database
- Logins are created before User.
- After you have logged into an instance(using your login account), you must have a user account in whichever database you want to use
- No user account? No access!

Login vs User

- Accessing the lobby of a building(Synonymous to an instance of in SQL Server) doesn't guarantee you accessing offices in the building
- Having a login isn't sufficient to access a database. You need a user account to access the database.

Securable

- A **securable** is anything that can have permissions granted or deny on in **SQL** Server. Example Table, view.
- If you need to have a GRANT to perform an action on an object, the object is a securable.
- A SQL Server instance contains a hierarchical collection of entities. Each instance contains multiple databases, and each database contains a collection of securable objects.
- Every SQL Server securable has associated *permissions* that can be granted to a *principal*, which is an individual, group or process granted access to SQL Server. The SQL Server security framework manages access to securable entities through *authentication* and *authorization*

AUTHENTICATION And AUTHORIZATION

- Authentication – verifying who you are.
- Authorization - verifying that you have access to something

- Authentication is the process of logging on to SQL Server by which a principal requests access by submitting credentials that the server evaluates. Authentication establishes the identity of the login or process being authenticated.

- Authorization is the process of determining which securable resources a principal can access, and which operations are allowed for those resources.

AUTHENTICATION(Login) TYPES

- There are two types of Authentication – Windows Authentication and SQL Server authentication
- WINDOWS Authentication - Domain\WindowsName
- When you are accessing SQL Server from the same computer it is installed on, you shouldn't be prompted to type in a username and password.
- The SQL Server service already knows that someone is logged in into the operating system with the Windows credentials, and it uses these credentials to allow the user into its databases. Of course, this works if the client resides on the same computer as the SQL Server, or if the connecting client matches the Windows credentials of the server.
- Windows authentication is generally more secured than SQL Server authentication

SQL Server Authentication

- Allows SQL Server to support older applications and applications provided by third parties that require SQL Server Authentication.
- Allows SQL Server to support environments with mixed operating system where all users are not authenticated by a Windows domain

Roles

- Allows you to assign permissions to a group of users instead of to individual users
- Logins and users can be grouped into roles
- Server role- allow you to assign a set of permissions to a group of logins across the server instance.
- Database role- allow you to assign a set of permissions to a group of users in a specific database.

Server Roles

- **Bulk Admin** - Members of the bulkadmin fixed server role can run the BULK INSERT statement.
- **Dbcreator** - Members of the dbcreator fixed server role can create, alter, drop, and restore any database.
- **Diskadmin** - The diskadmin fixed server role is used for managing disk files
- **Public** - Every SQL Server login belongs to the public server role. When a server principal has not been granted or denied specific permissions on a securable object, the user inherits the permissions granted to public on that object. Only assign public permissions on any object when you want the object to be available to all users. You cannot change membership in public.
- **Securityadmin** - Members of the securityadmin fixed server role manage logins and their properties. They can GRANT, DENY, and REVOKE server-level permissions. They can also GRANT, DENY, and REVOKE database-level permissions if they have access to a database. Additionally, they can reset passwords for SQL Server logins.The **securityadmin** role should be treated as equivalent to the **sysadmin** role.

- **Serveradmin** - Members of the serveradmin fixed server role can change server-wide configuration options and shut down the server.

- **Sysadmin** - Members of the sysadmin fixed server role can perform any activity in the server.

Database role

- Fixed database roles - pre-defined set of permissions that are designed to allow you to easily manage groups of permissions.
- User-defined database roles – roles created by a database administrator.

Database Roles

- **db_owner:** Members have full access in a database.
- **db_datareader:** Members can read all data.
- **db_datawriter:** Members can add, delete, or modify data in the tables.
- **db_securityadmin:** Members can modify role membership and manage permissions.
- **db_bckupoperator:** Members can back up the database.

Schema

- Schema is a collection of database objects that are grouped together and share the same name resolution (tables, view, SPs, functions etc).
- Eg – HR.Employee, HR.Address, HR.Salary for tables and view that belong to Human resource team rather than using the generic dbo schema.
- Schema help in securing database objects because it will be easier to identify objects that belong to one business unit or group.
- The default schema is dbo.

Data breach

- A data breach is when secured information is released to an untrusted environment.
- What causes a data breach:
- Hacking by external group
- Malicious user on the inside
- Software bug
- Backup files being exposed
- Theft of a computer that holds sensitive data

Orphaned Users

- Orphaned users in SQL Server occur when a database user is based on a login in the **master** database, but the login no longer exists in **master**.
- This can occur when the login is deleted, or when the database is moved to another server where the login does not exist.

Exercise 1

- Create a new SQL Server login (Sholla101) and create a corresponding user with the same name in Movies database
- Make the default schema for the account dbo.
- Make the new account part of the **db_datareader** group

Provide the following access in Movies database.

- SELECT on tblActor table
- INSERT, UPDATE and SELECT on tblCast table
- SELECT on vwFilms view
- SELECT and ALTER on vwFilmSimple view.

Exercise 2

- Create two SQL Server Logins (SQLUser101 and SQLUser102). Add this two logins as a user in Movies database.
- Create a DB Role called DBSuperUsers and add SQLUser101 and SQLUser102 as members. Grant the new user SELECT, INSERT and DELETE permission on tblDirector and tblActor database