Laboratorio de seguridad

Tabla de contenidos

[Configure las propiedades de Security del SQL Server, para permitir los posibles métodos de autenticación 2](#_Toc474400683)

[Identifique las diferencias existentes entre cada tipo de autenticación y la política de complejidad y expiración de contraseñas. 3](#_Toc474400684)

[Crear cuentas en forma correcta 5](#_Toc474400685)

[Autenticación SQL Server 5](#_Toc474400686)

[Autenticación de dominio de Windows 13](#_Toc474400687)

[Excepciones 22](#_Toc474400688)

[Genera el reporte detallado e interpreta los resultados. 25](#_Toc474400689)

[Roles a nivel de servidor por usuario 25](#_Toc474400690)

[Roles a nivel de base de datos por usuario 25](#_Toc474400691)

[Permisos de usuario a nivel de objetos de una base de datos 26](#_Toc474400692)

[Conclusiones 28](#_Toc474400693)

[Recomendaciones 29](#_Toc474400694)

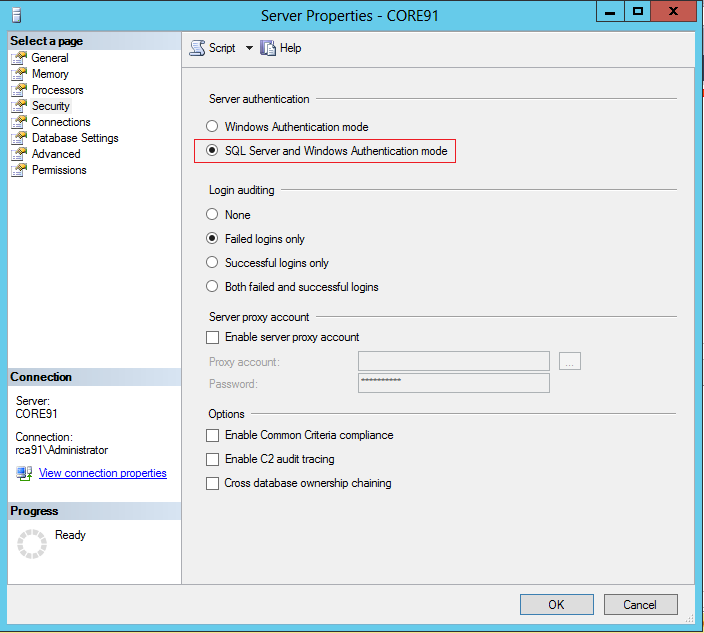
[Bibliografía 30](#_Toc474400696)

## Configure las propiedades de Security del SQL Server, para permitir los posibles métodos de autenticación

SQL Server ofrece dos métodos de autenticación:

* Windows Authentication mode; utiliza cuentas del dominio de directorio activo (Active Directory).
* SQL Authentication mode; permite la configuración de cuentas directamente administradas por el SQL Server, independientemente de su existencia o no en el dominio activo. La configuración de las cuentas, las contraseñas y sus permisos, recaen directamente sobre el administrador de la base de datos.

Para seleccionar entre ellas, se utiliza el tab de preferencias del servidor. Se puede escoger entre utiliza exclusivamente autenticación basada en windows o habilitar también la autenticación de SQL Server, cómo puede verse en la siguiente interfaz.



¿Cómo habilitar la cuenta sa?

Durante la configuración estándar de las ediciones más recientes de SQL server esta cuenta se encuentra deshabilitado. Los siguientes comandos de TSQL permiten activarlas y definir una contraseña.

ALTER LOGIN sa ENABLE;

ALTER LOGIN sa WITH PASSWORD = 'Pa$$word';

## Identifique las diferencias existentes entre cada tipo de autenticación y la política de complejidad y expiración de contraseñas.

|  |  |
| --- | --- |
| **Autenticación SQL Server** | **Autenticación Windows** |
| El usuario debe usar un usuario/contraseña para entrar al servidor, y luego un usuario / contraseña distinto para conectarse a SQL Server | El usuario puede conectarse a SQL Server usando las credenciales del usuario de la sesión activa de Windows en el momento en que se establece la conexión |
| Intercambio de handshake y challenge menos seguro, al no poder utilizar el protocolo de seguridad Kerberos. | Utiliza el protocolo de seguridad Kerberos para establecer la autenticación. |
| No ofrece todas las políticas de seguridad que ofrece los Dominios de Windows, pero usa la configuración local del servidor para manejar aspectos como la expiración de contraseñas y así como las políticas de contraseñas disponibles desde Windows Server 2003. | Permite definir políticas de complejidad de contraseñas y expiración de contraseñas que pueden ser aplicadas a uno o más usuarios del dominio, de forma consistente. |
| La contraseña encriptada de SQL Server debe ser transmitida por la red al realizar la conexión. Algunas aplicaciones que se conectan automáticamente ofrecen la posibilidad de guardar la contraseña, lo cual crea el riesgo local de que dicha contraseña se almacena de forma no encriptada. Ejemplo, conexiones de Excel 2007 donde el usuario almacena la contraseña en el odt. La contraseña almacena no es encriptada. | SQL Server no pide la contraseña. La validación de la identidad ocurre a través de tokens y la garantía de que la sesión fue previamente autorizada por el servidor de dominio. |
| Permite soportar aplicaciones heredadas (legacy), por ejemplo aquellas aplicaciones que solamente aceptan autenticación de SQL Server. | No se puede utilizar con algunas aplicaciones heredadas. |
| Soporta ambientes con sistemas operativos mixtos, cuando no todos los usuarios están autenticados por el dominio de Windows. | Solo funciona con Windows |

|  |  |
| --- | --- |
| **Autenticación SQL Server** | **Autenticación Windows** |
| Permite que los usuarios se conecten desde dominios desconocidos o no certificados. | Solo se pueden conectar desde dominios de confianza. |
| Permite el soporte a aplicaciones web donde los usuarios pueden crear sus propias identidades. | Los usuarios no pueden crear sus propias identidades, hasta deben ser creadas previamente en los servidores de dominio. |

## 

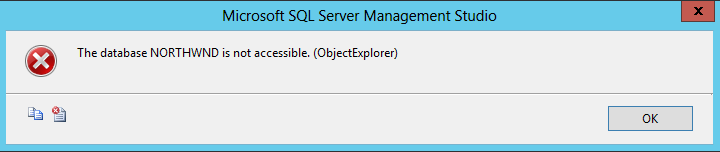
## Crear cuentas en forma correcta

### Autenticación SQL Server

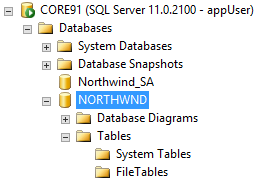
#### T-SQL

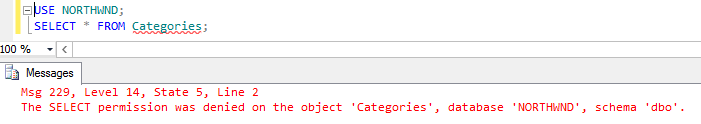
1. Crear un LOGIN con autenticación SQL  
     
   CREATE LOGIN appUser

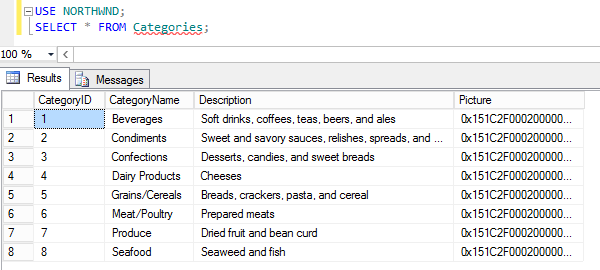
WITH PASSWORD = 'Pa$$word'

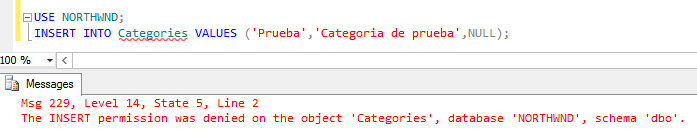
1. Asignar rol de servidor  
     
   ALTER SERVER ROLE diskadmin ADD MEMBER appUser;
2. Crear un USER para la base de datos NORTHWND  
     
     
     
   USE NORTHWND;

CREATE USER appUser FROM LOGIN appUser;

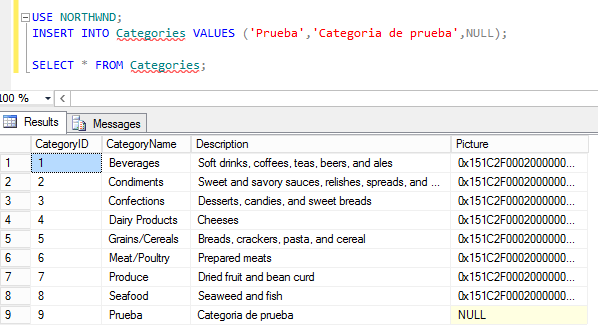


1. Asignar rol de base de datos  
     
     
     
   USE NORTHWND;

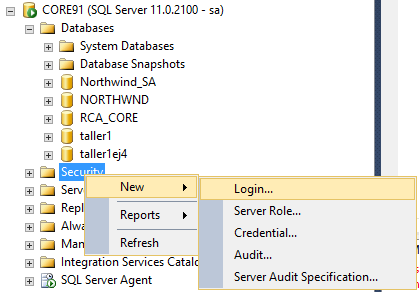
ALTER ROLE db\_datareader ADD MEMBER appUser;  
  


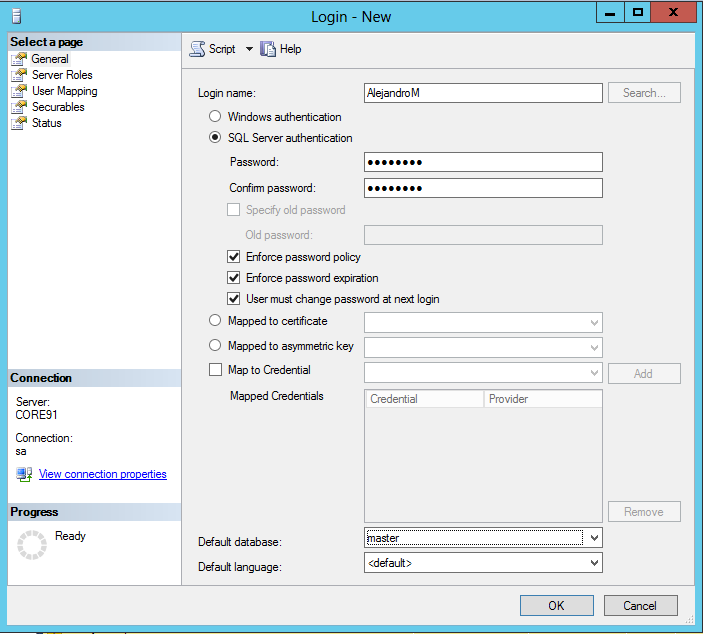
1. Asignar privilegios  
     
     
     
   USE NORTHWND;

GRANT INSERT ON dbo.Categories TO appUser;

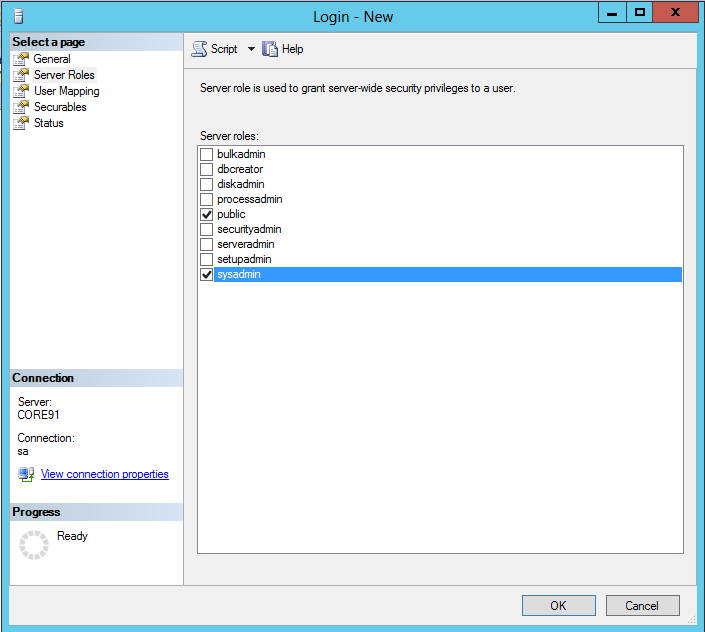


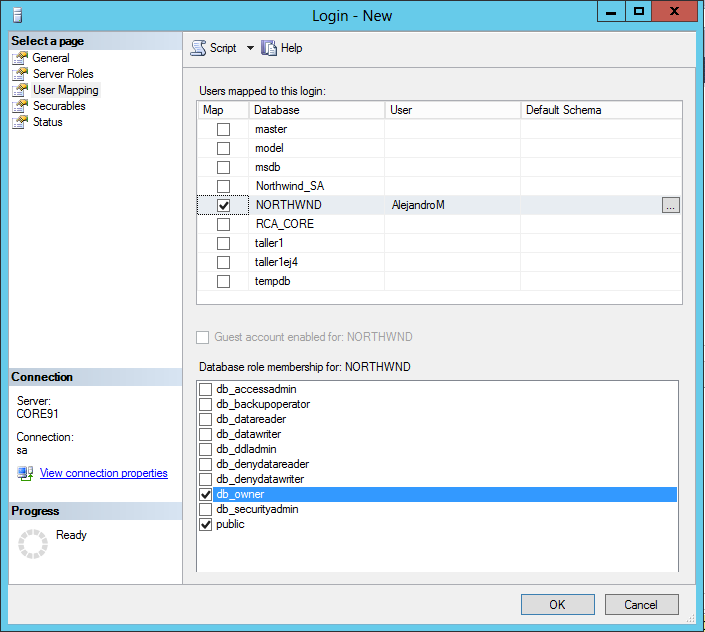
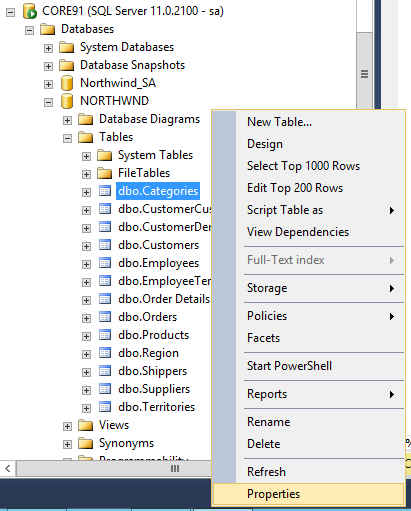
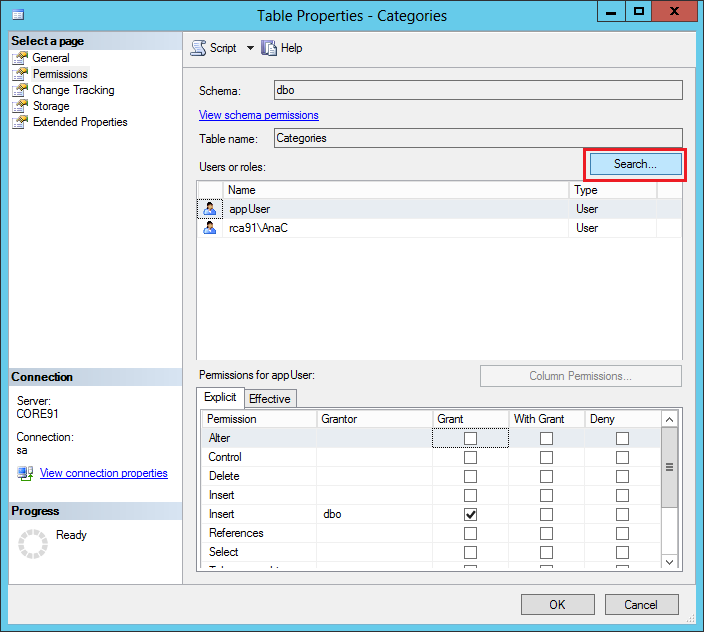
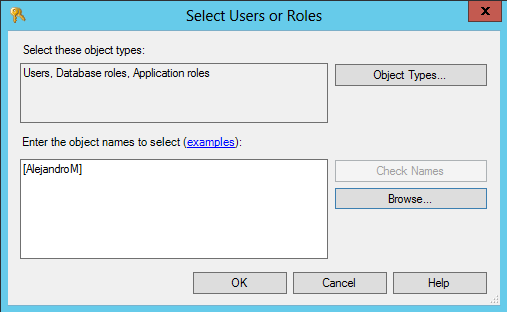
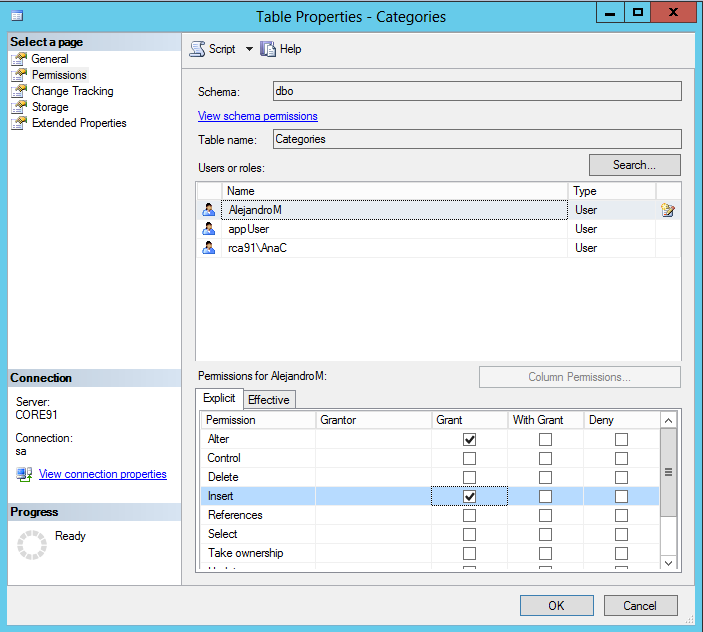
#### Utilizando GUI

1. Crear un LOGIN con autenticación SQL  
     
   



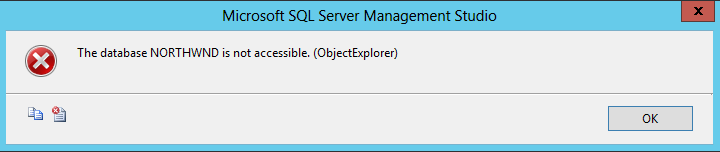
1. Asignar el rol fijo de servidor pertinente a la cuenta de login creada.



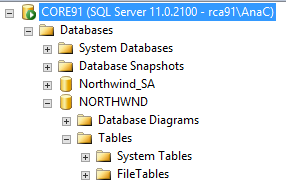
1. Crear un USER para la base de datos NORTHWND y asignar rol de base de datos db\_owner.  
     
   
2. Asignar privilegios  
     
     
     
     
     
     
     
   

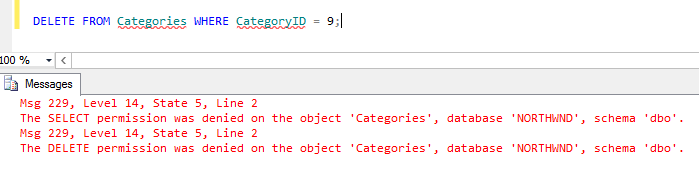
### Autenticación de dominio de Windows

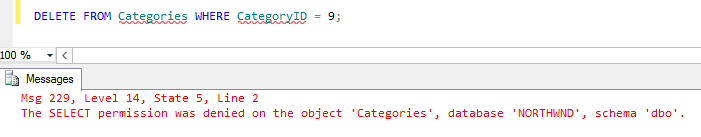
#### T-SQL

1. Crear un LOGIN con autenticación Windows  
     
   CREATE LOGIN [rca91\AnaC] FROM WINDOWS;
2. Asignar rol de servidor  
     
   ALTER SERVER ROLE bulkadmin ADD MEMBER [rca91\AnaC];
3. Crear un USER para la base de datos NORTHWND  
     
     
     
   USE NORTHWND

CREATE USER [rca91\AnaC] FOR LOGIN [rca91\AnaC]

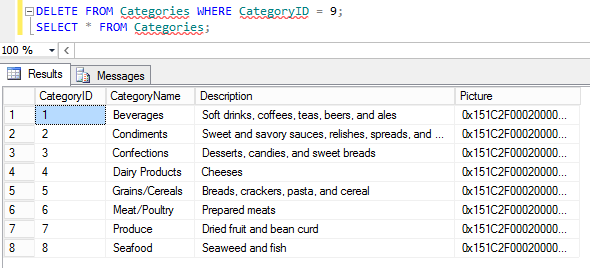


1. Asignar rol de base de datos  
     
     
     
   USE NORTHWND;

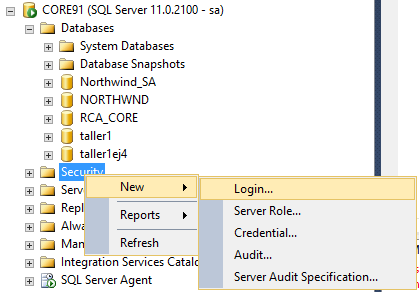
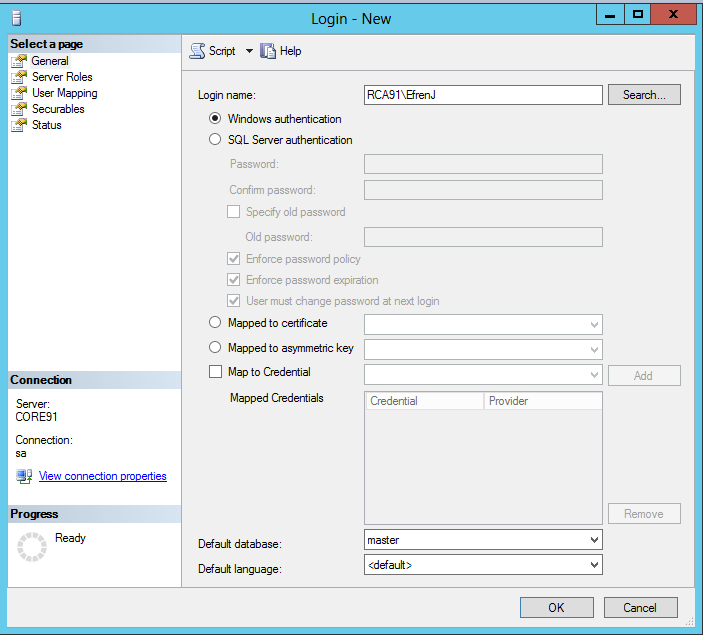
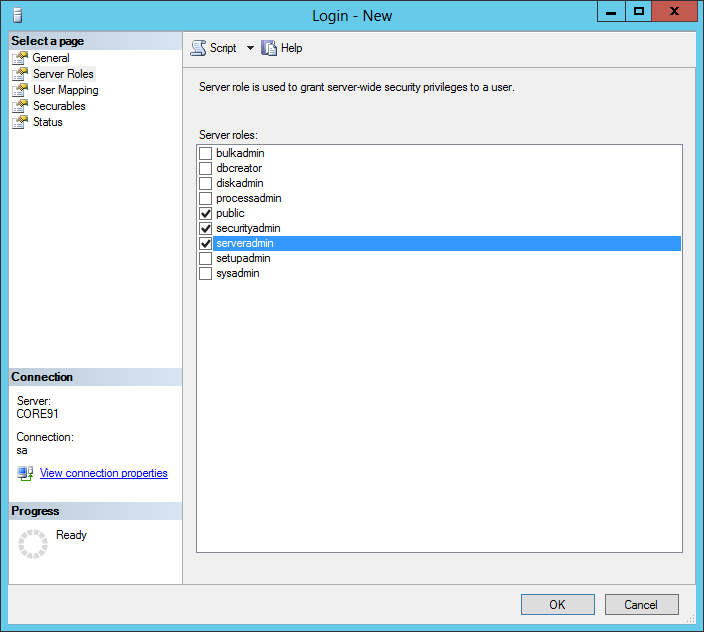
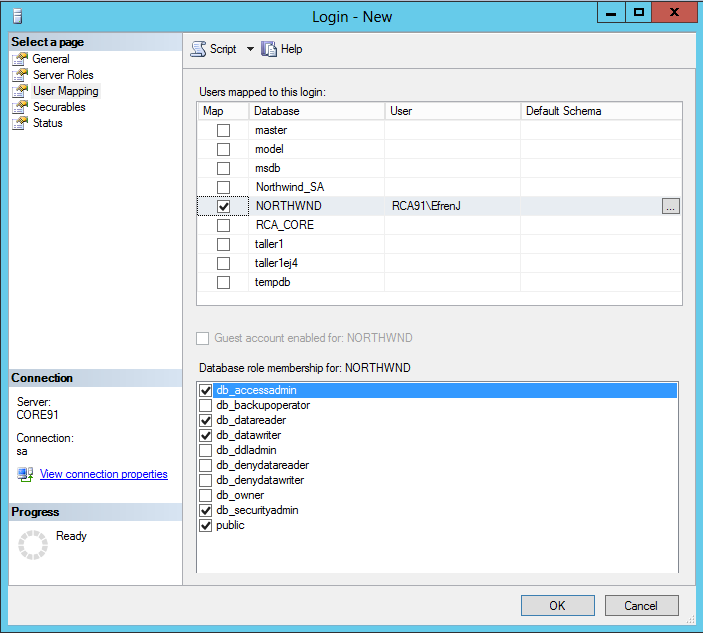
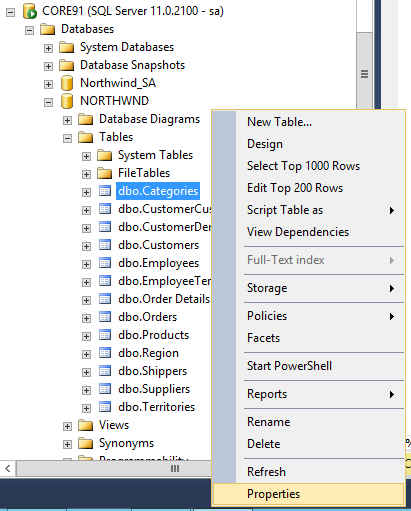
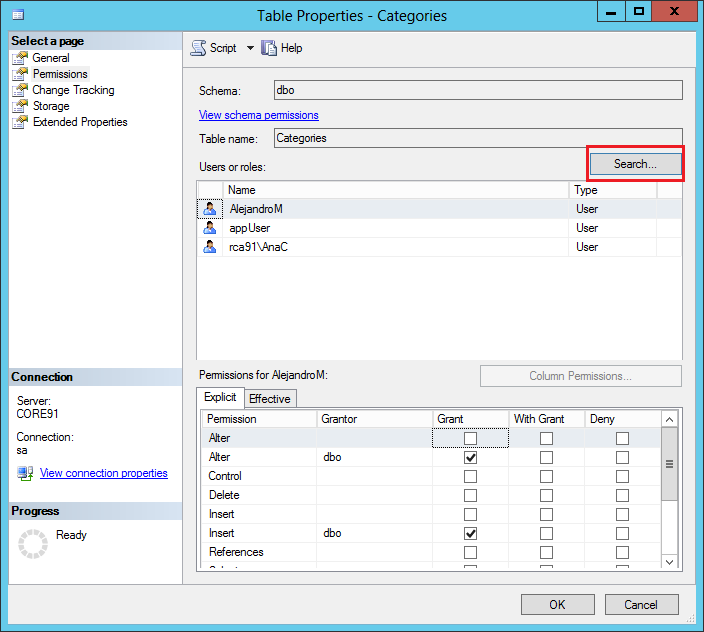
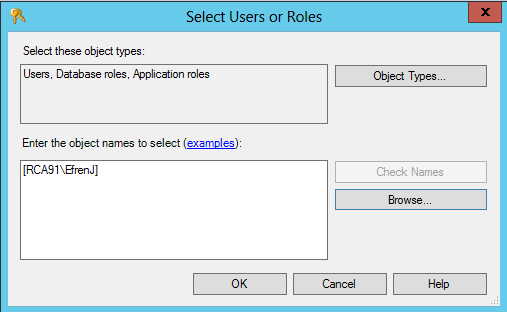
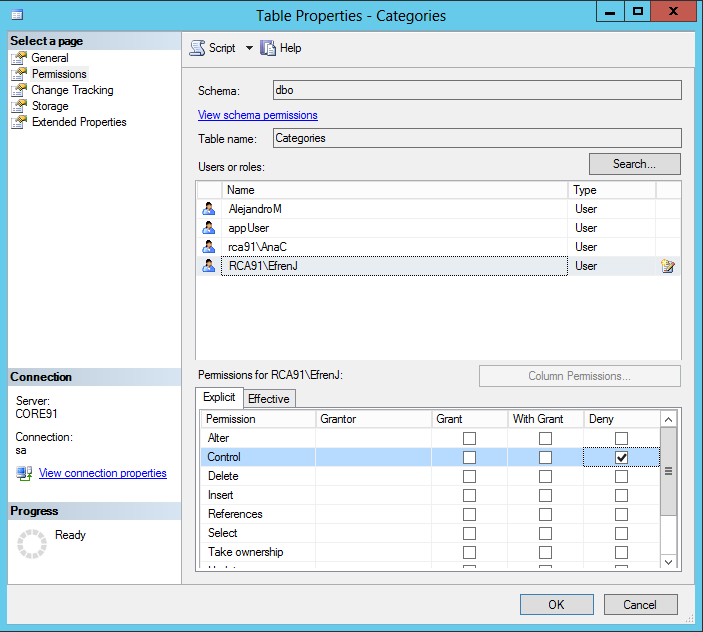
ALTER ROLE db\_datawriter ADD MEMBER [rca91\AnaC];  
  


1. Asignar privilegios  
     
   USE NORTHWND;

GRANT SELECT ON dbo.Categories TO [rca91\AnaC];

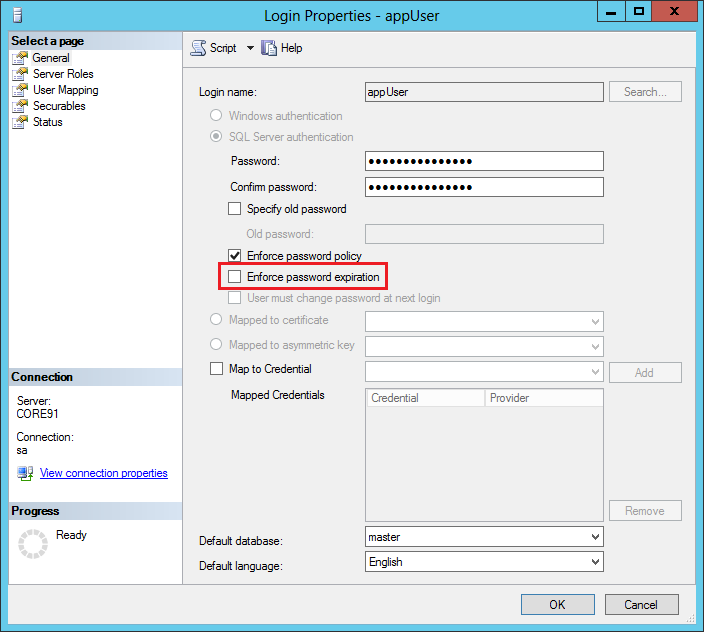
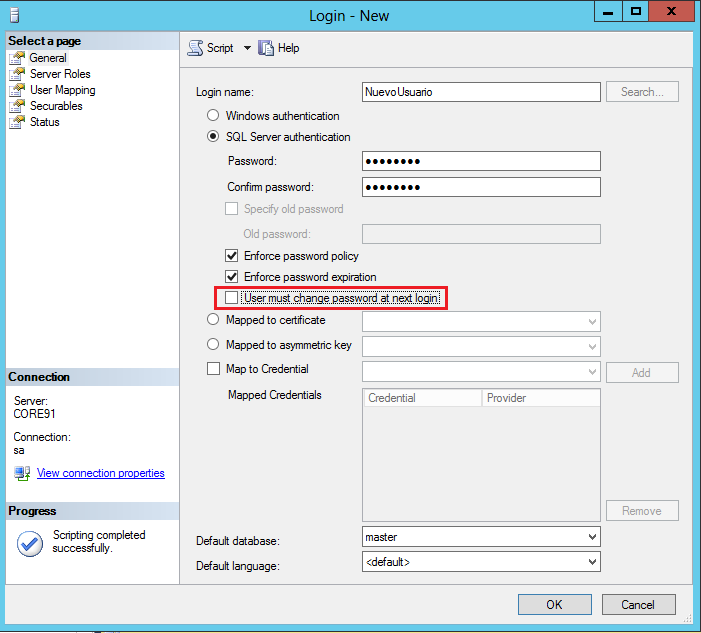


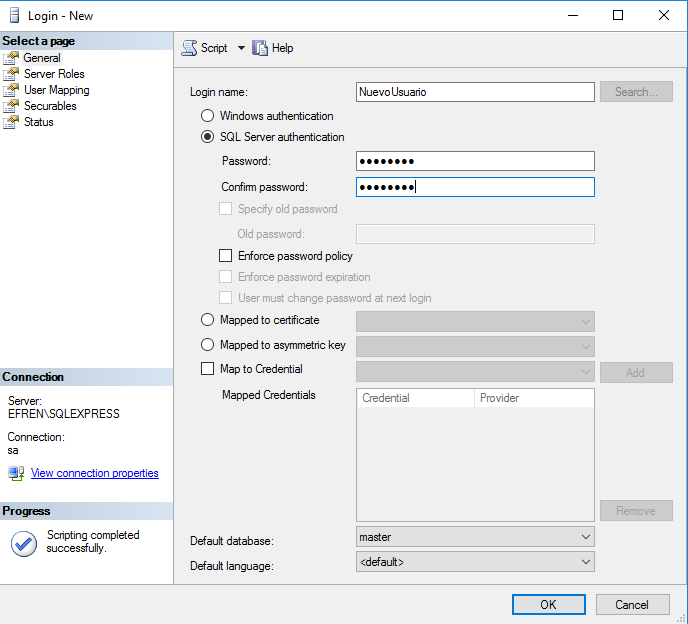
#### Utilizando GUI

1. Crear un LOGIN con autenticación Windows  
     
     
     
   
2. Asignar rol de servidor  
     
   
3. Crear un USER para la base de datos NORTHWND y asignar roles de base de datos  
     
   
4. Asignar privilegios  
     
     
     
     
     
     
     
     
   

## 

## Excepciones

1. Deshabilitar expiración de contraseña  
     
   En algunos casos puede ser necesaria la deshabilitación de la expiración de la contraseña de un Login de SQL. Por ejemplo, si la combinación de usuario y contraseña es usada por una aplicación externa y la contraseña expira, la aplicación no podría acceder a la base de datos hasta que se realice el cambio de contraseña del lado de la aplicación. En este caso, es posible que sea mejor deshabilitar la expiración de la contraseña para que no suceda.  
     
   T-SQL:  
     
   ALTER LOGIN [appUser] WITH CHECK\_EXPIRATION=OFF  
     
   GUI:  
     
   
2. El usuario no debe cambiar la contraseña la próxima vez que haga login  
     
   Si la contraseña debe ser compartida, guardada o recordada por varias personas, o si la próxima persona que ingrese utilizando el nuevo login no será la persona que utilizará la cuenta todos los días, entonces es preferible deshabilitar la opción de “cambiar la contraseña la próxima vez que haga login”.  
     
   T-SQL  
     
   **/\*Si no se especifica la opción “MUST\_CHANGE”, entonces no será necesario cambiar la contraseña\*/**  
   CREATE LOGIN [NuevoUsuario] WITH PASSWORD=N'Pa$$word' --MUST\_CHANGE  
     
   GUI  
   
3. El usuario no debe tener políticas de contraseña  
     
   Si debe existir algún usuario a el cual no se le debe solicitar políticas de seguridad debido a conexiones existentes con sistemas ya sean de respaldos o de auditoría, y estos demanden mantener conexiones y contraseñas constantes debido a su uso automático se plantea la no utilización de políticas de contraseña.  
     
   T-SQL  
     
   **/\*No se debe especifica la opción “**CHECK\_EXPIRATION**” y “**CHECK\_POLICY**”, entonces no será necesario utilizar políticas de contraseña\*/**  
   CREATE LOGIN [NuevoUsuario] WITH PASSWORD=N'Pa$$word' CREATE LOGIN [NuevoUsuario] PASSWORD=N'Pa$$word', CHECK\_EXPIRATION=OFF, CHECK\_POLICY=OFF  
     
   GUI



## Genera el reporte detallado e interpreta los resultados.

### Roles a nivel de servidor por usuario

SELECT

a.name as Name

, a.type\_desc AS LoginType

, ISNULL(SUSER\_NAME(b.role\_principal\_id),'public') AS AssociatedServerRole

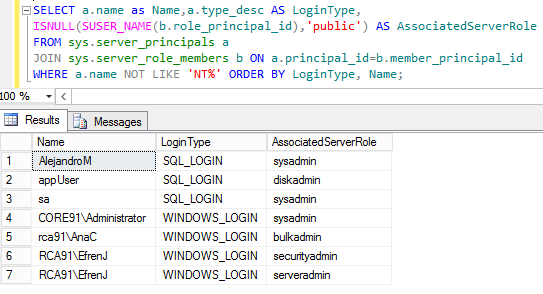
FROM sys.server\_principals a

JOIN sys.server\_role\_members b

ON a.principal\_id=b.member\_principal\_id

WHERE a.name NOT LIKE 'NT%'

ORDER BY LoginType, Name;



### Roles a nivel de base de datos por usuario

SELECT

pr.name AS UserName

, pr.type\_desc AS LoginType

, USER\_NAME(me.role\_principal\_id) AS AssociatedDatabaseRole

, DB\_NAME() AS 'Database'

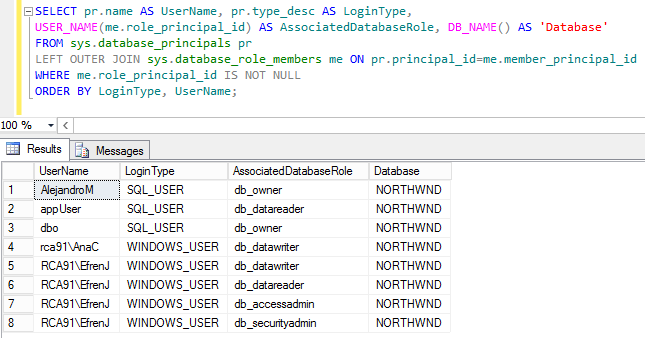
FROM sys.database\_principals pr

LEFT OUTER JOIN sys.database\_role\_members me

ON pr.principal\_id=me.member\_principal\_id

WHERE me.role\_principal\_id IS NOT NULL

ORDER BY LoginType, UserName;



### Permisos de usuario a nivel de objetos de una base de datos

SELECT

pr.type\_desc AS LoginType

, pr.name as Name

, pe.permission\_name AS 'Action',pe.state\_desc AS 'Permission'

, CASE class

WHEN 0 THEN 'Database::' + DB\_NAME()

WHEN 1 THEN OBJECT\_NAME(major\_id)

WHEN 3 THEN 'Schema::' + SCHEMA\_NAME(major\_id)

END AS 'Securable'

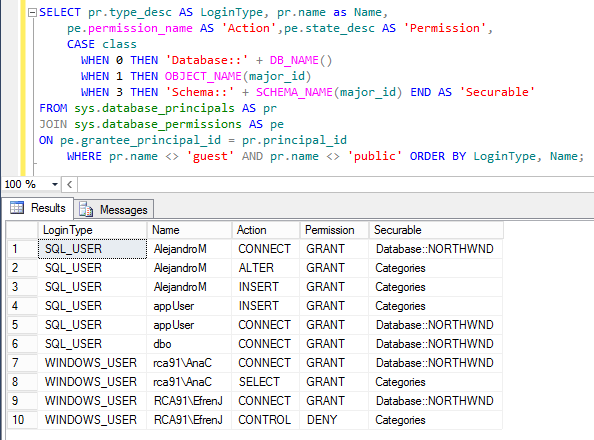
FROM sys.database\_principals AS pr

JOIN sys.database\_permissions AS pe

ON pe.grantee\_principal\_id = pr.principal\_id

WHERE pr.name <> 'guest' AND pr.name <> 'public'

ORDER BY LoginType, Name;



## 

## Conclusiones

1. El nivel de integración que proporciona el Microsoft SQL Server con las tecnologías de autenticación empresarial del mismo fabricante, permite una experiencia transparente en el proceso de conexión con los usuarios. Es muy clara la división de las siguientes funciones:
   1. Creación de usuarios, mantenimiento de contraseñas (Active Directory).
   2. Habilitar la conexión de usuarios existentes con cuentas de la base de datos. Se observa que el uso de cuentas LOGIN y cuentas USER es el resultado de soportar el sistema de autenticación basado en SQL SERVER, que no deja de ser un requisito estándar de las implementaciones de SQL (SQL Server).
   3. Administración de los permisos de acceso de los usuarios a los componentes de la base de datos (SQL Server).
2. La existencia de la autenticación independiente de los usuarios (Microsoft SQL Authentication) no debe verse simplemente como el resultado de utilizar tecnologías más antiguas, sino también con la necesidad de soportar la conectividad y autenticación cuando se trata de clientes distintos de la cartera de productos de Microsoft, por ejemplo:
   1. A nivel de sistemas operativos diferentes como Linux, UNIX.
   2. A nivel de productos no provistos por Microsoft en estos ambientes, donde se requiere autenticación mediante ODBC, JDBC, etc.
3. La asignación de permisos en SQL Server no solamente permite la granularidad requerida por los estándares de SQL (usuario, tabla), sino también extender los conceptos de grupo del dominio activo, para simplificar y normalizar el manejo a lo largo de una organización. Esto permite autorizar o denegar a los usuarios diferentes funciones y accesos a nivel de tabla y de base de datos. Estos permisos se pueden asignar a ambos tipos de usuario, o pueden ser heredados según los roles que le han sido asignados.

## 

## Recomendaciones

1. La fase de diseño de los diferentes accesos en SQL Server requiere un análisis previo de las prácticas de la organización, para diseñar apropiadamente aquellas políticas que mejor se adapten a estos requerimientos. Por ejemplo, algunas organizaciones, pueden poseer una estructura bastante plana cuando se trata de las unidades de desarrollo y soporte de sus bases de datos, un solo grupo de TI puede estar a cargo de dicha organización. Otras, por su parte, pueden por el contrario tener una gran cantidad de grupos de negocio y sus áreas de soporte pueden encontrarse alineadas a dichas unidades de negocio o pueden tener más bien tener una organización matricial. En estos casos, diseños una estructura de seguridad, requerirá el manejo de un mayor número de jerarquías y niveles de acceso, así como conceptos de cuentas de servicio.
2. Uno de los retos más importantes que algunas organizaciones pueden enfrentar, es la designación de las cuentas de usuario bajo las cuales ejecutan los procesos críticos, como el SQL server. En el pasado, dos modelos básicos han operado:
   1. Utilizar cuentas locales en los servidores de base de datos, bajo las cuales los procesos principales de SQL server son ejecutados. Esto tiene la desventaja de la falta de centralización de las políticas de ejecución y la necesidad de mantener la renovación de contraseñas en cada equipo individual.
   2. Crear cuenta de pseudo usuario, cuyo único propósito es la ejecución de los servicios de SQL server en los servidores. SI bien, esto elimina el problema de la descentralización, no obstante, se queda corto cuando se trata de manejar la vida de las contraseñas y limitar la ejecución de tales o cuales servicios a los servidores previamente definidos.
   3. La aparición en Windows 2008 R2 y el subsiguiente desarrollo en 2012, de las cuentas globales de administración, o cuentas grupales de administración, vienen a atacar de frente este problema. Con este modelo, es posible crear usuarios cuyo único propósito es correr servicios en uno o más equipos previamente definidos. El manejo de la vida de las contraseñas recae sobre el sistema de directorio activo, sin requerir la intervención del administrador. De tal forma, se obtiene lo mejor de ambos mundos.
3. Las configuraciones por defecto del base de datos toman en cuenta una serie de compromisos entre la seguridad y la operación normal de los sistemas en las bases instaladas de los clientes, especialmente consideraciones de soporte de ediciones anteriores o productos populares. Por este motivo, muchas veces las decisiones de configuración por defecto no pueden asumirse como “seguras”, especialmente si se trata de sistemas dónde la exposición de la información acarrea riesgos sobre los datos financieros de los clientes. Debido a esto, una de las tareas del administrador y los grupos de seguridad, es evaluar las configuraciones por defecto de forma detallada, definir el nivel de apetito de riesgo de sus compañías y los recursos materiales con los cuáles cuentan, para tomar las decisiones de configuración que mejor se alineen con su realidad.

## Bibliografía

ALTER ROLE (Transact-SQL). (s.f.). Recuperado de <https://msdn.microsoft.com/en-us/library/ms189775(v=sql.110).aspx>

ALTER SERVER ROLE (Transact-SQL). (s.f.). Recuperado de <https://msdn.microsoft.com/en-us/library/ee677634(v=sql.110).aspx>

Auditing SQL Server User and Role Permissions for Databases. (s.f.). Recuperado de <https://www.mssqltips.com/sqlservertip/2132/auditing-sql-server-user-and-role-permissions-for-databases/>

Choose an Authentication Mode. (s.f.). Recuperado de <https://msdn.microsoft.com/en-us/library/ms144284.aspx>

Change Server Authentication Mode. (s.f.). Recuperado de <https://msdn.microsoft.com/en-us/library/ms188670.aspx>

Create a Login. (s.f.). Recuperado de <https://msdn.microsoft.com/en-us/library/aa337562(v=sql.110).aspx>

Permissions of Fixed Database Roles (Database Engine). (s.f.). Recuperado de <https://msdn.microsoft.com/library/ms189612.aspx>

Permissions of Fixed Server Roles (Database Engine). (s.f.). Recuperado de <https://msdn.microsoft.com/library/ms175892.aspx>

Shyamsundar, A. (2014). Managed Service Accounts (MSA) and SQL 2012: Practical Tips. Recuperado de <http://blogs.msdn.com/b/arvindsh/archive/2014/02/03/managed-service-accounts-msa-and-sql-2012-practical-tips.aspx>

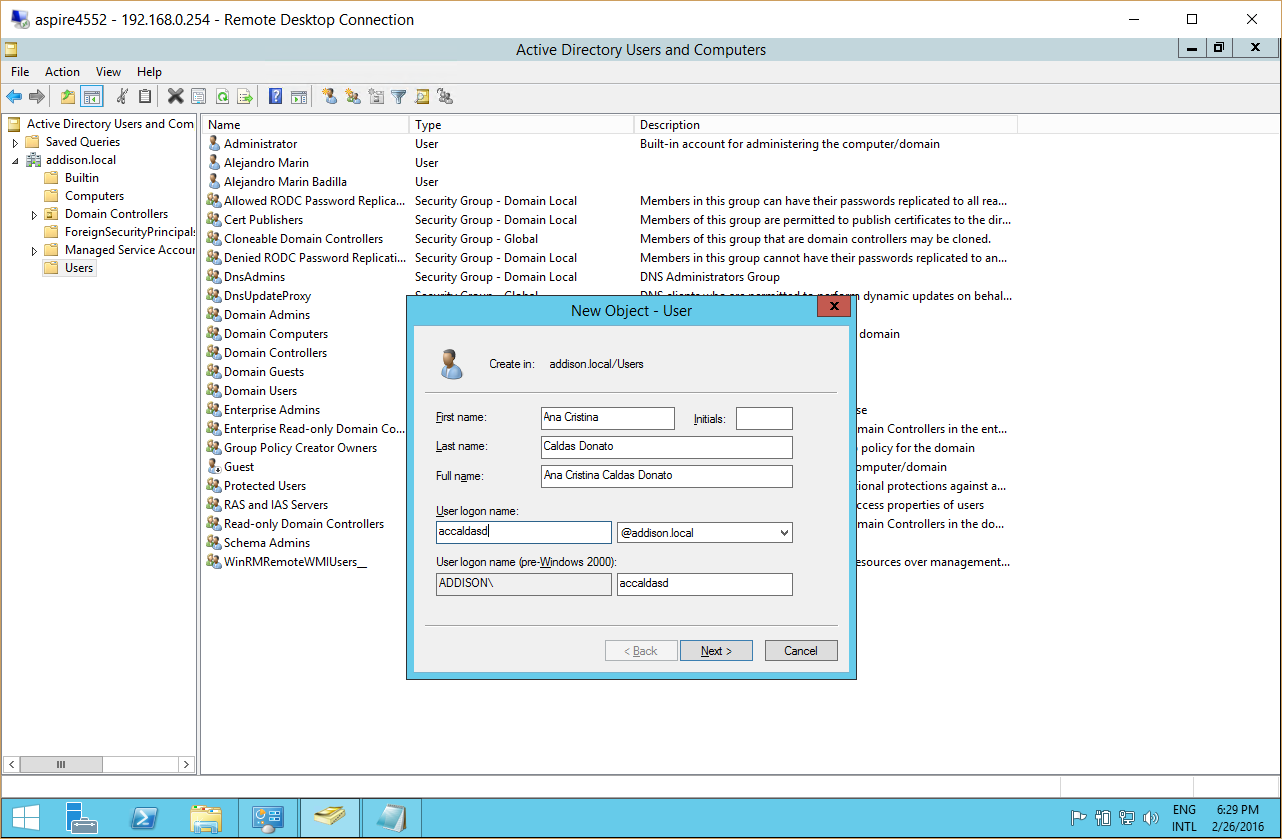
T-SQL: Retrieve all users and associated roles for ALL databases. (2012). Recuperado de <https://www.pythian.com/blog/httpconsultingblogs-emc-comjamiethomsonarchive20070209sql-server-2005_3a00_-view-all-permissions-_2800_2_2900_-aspx/>

Anexo 1 - Ejemplo de Creación de cuentas desde el Active Directory y autorización de SQL Server paso a paso

Configuración de los usuarios

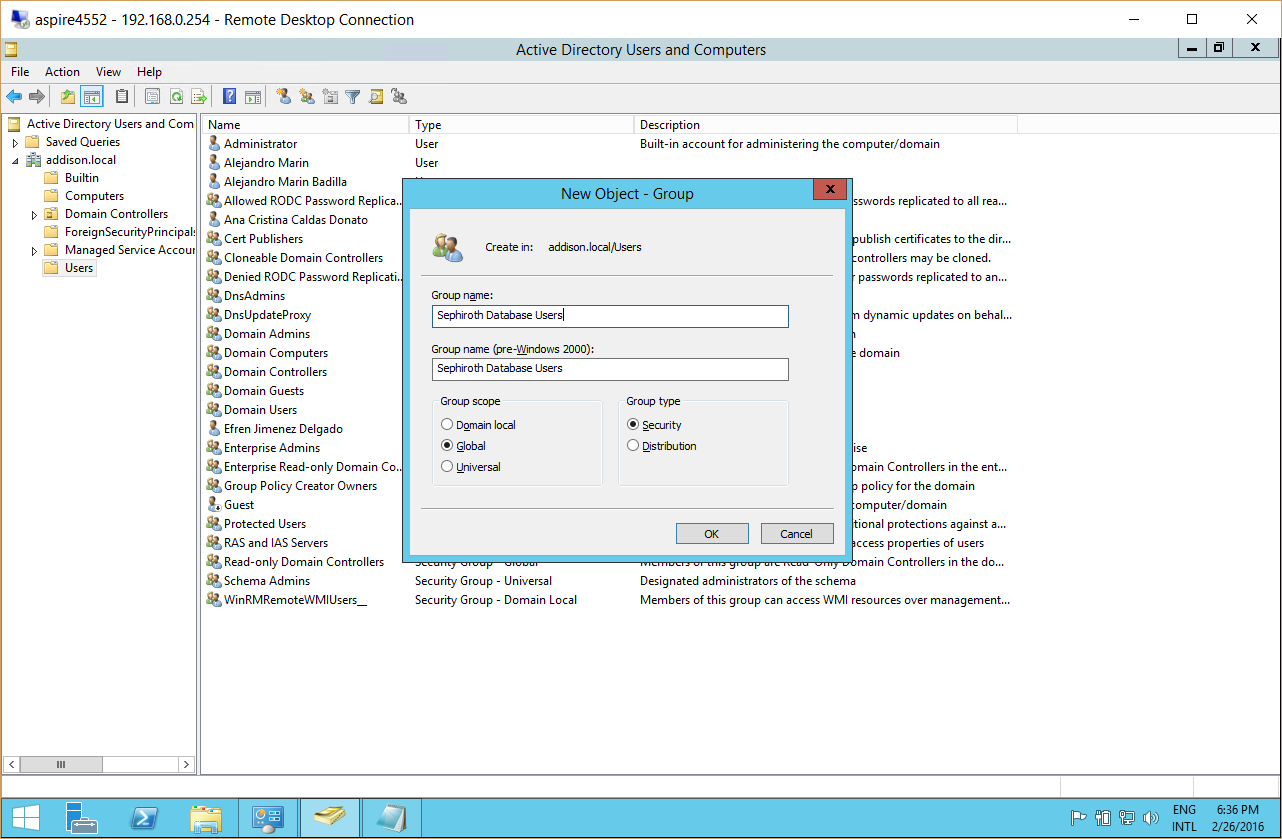
1. Crear el usuario en el Dominio Activo.

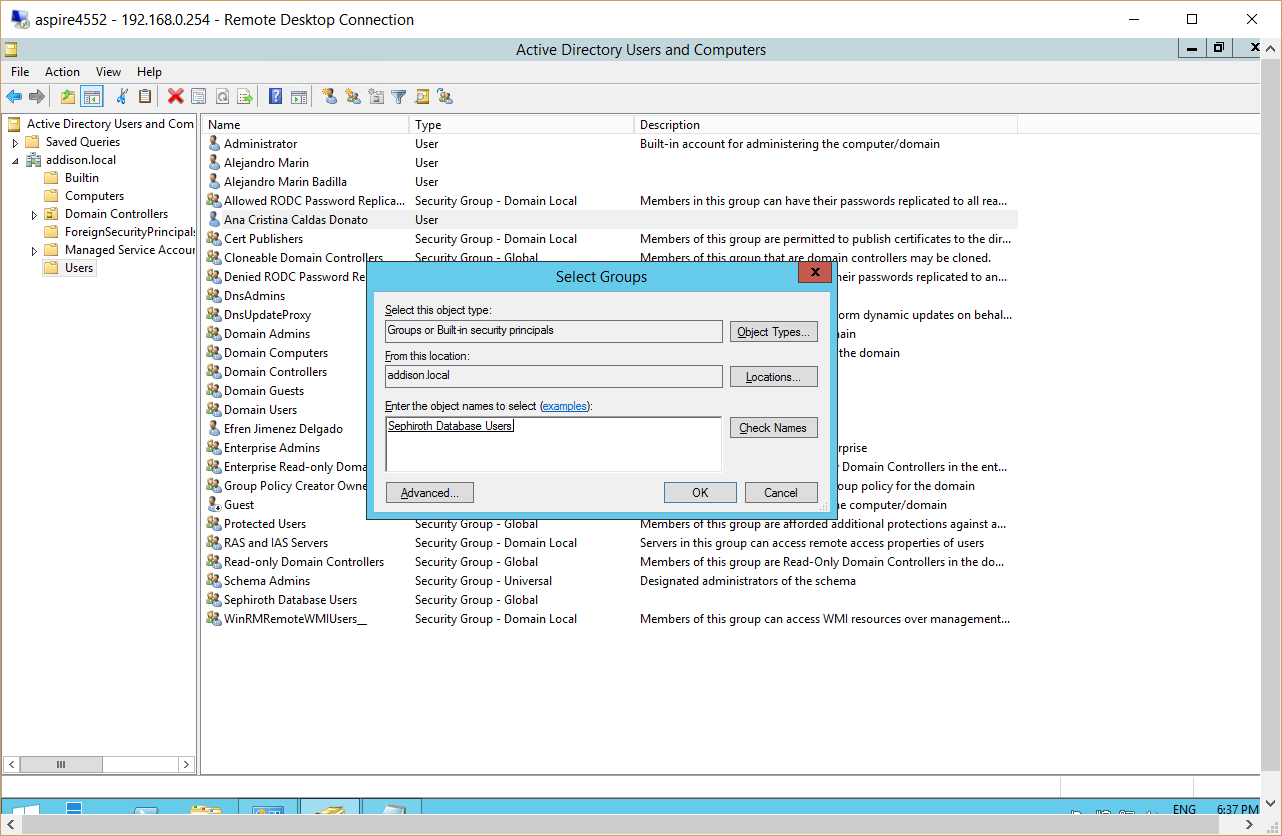
Por ejemplo, crear el usuario accaldasd, asociado a la persona Ana Cristina Caldas Donato, con la cuenta de correo caldas.donato@gmail.com.



2. Crear un Grupo para contener a los usuarios de la base de datos, y asignar los usuarios al grupo

Por ejemplo, crear el grupo “Sephiroth Database Users”

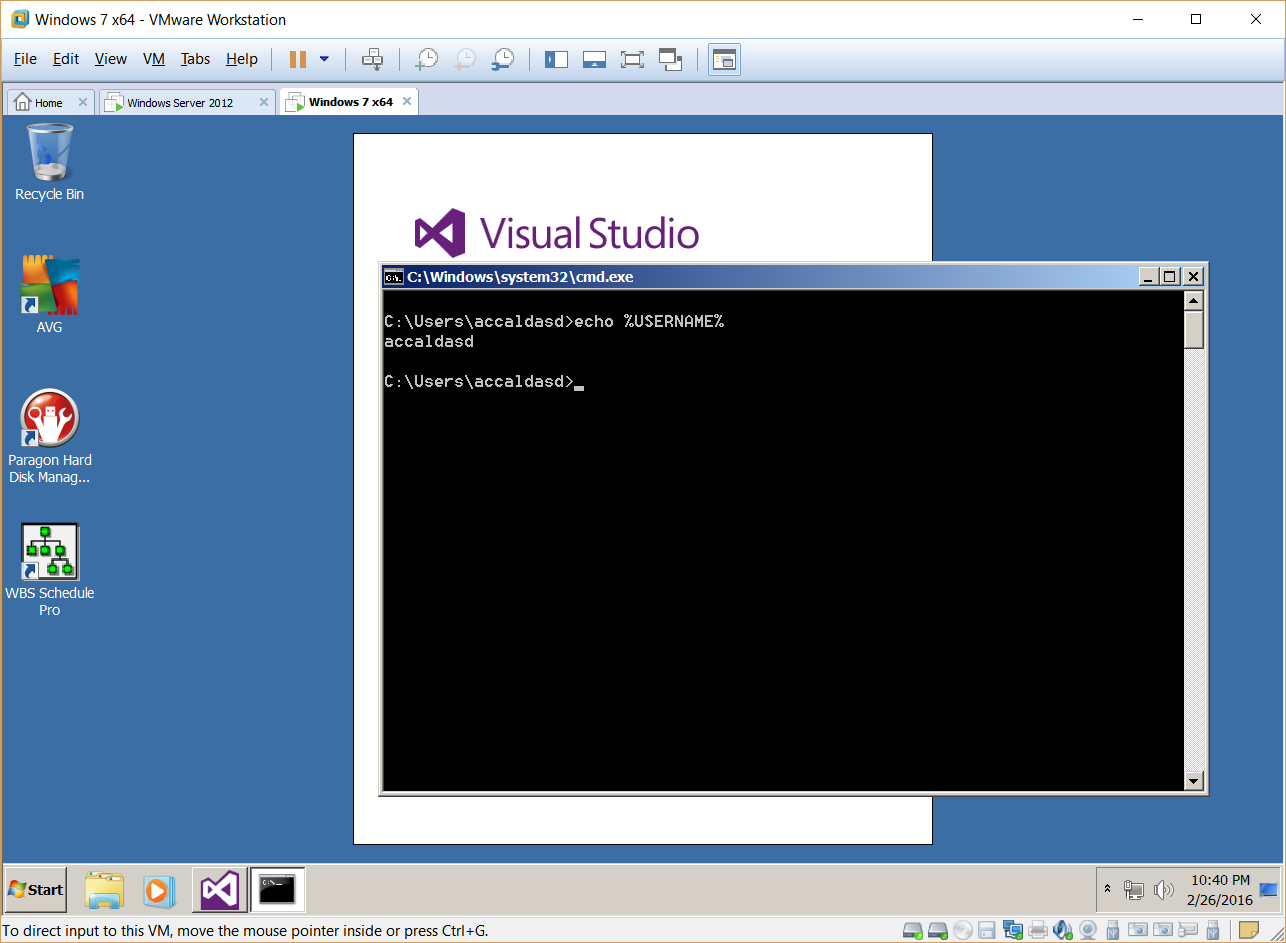


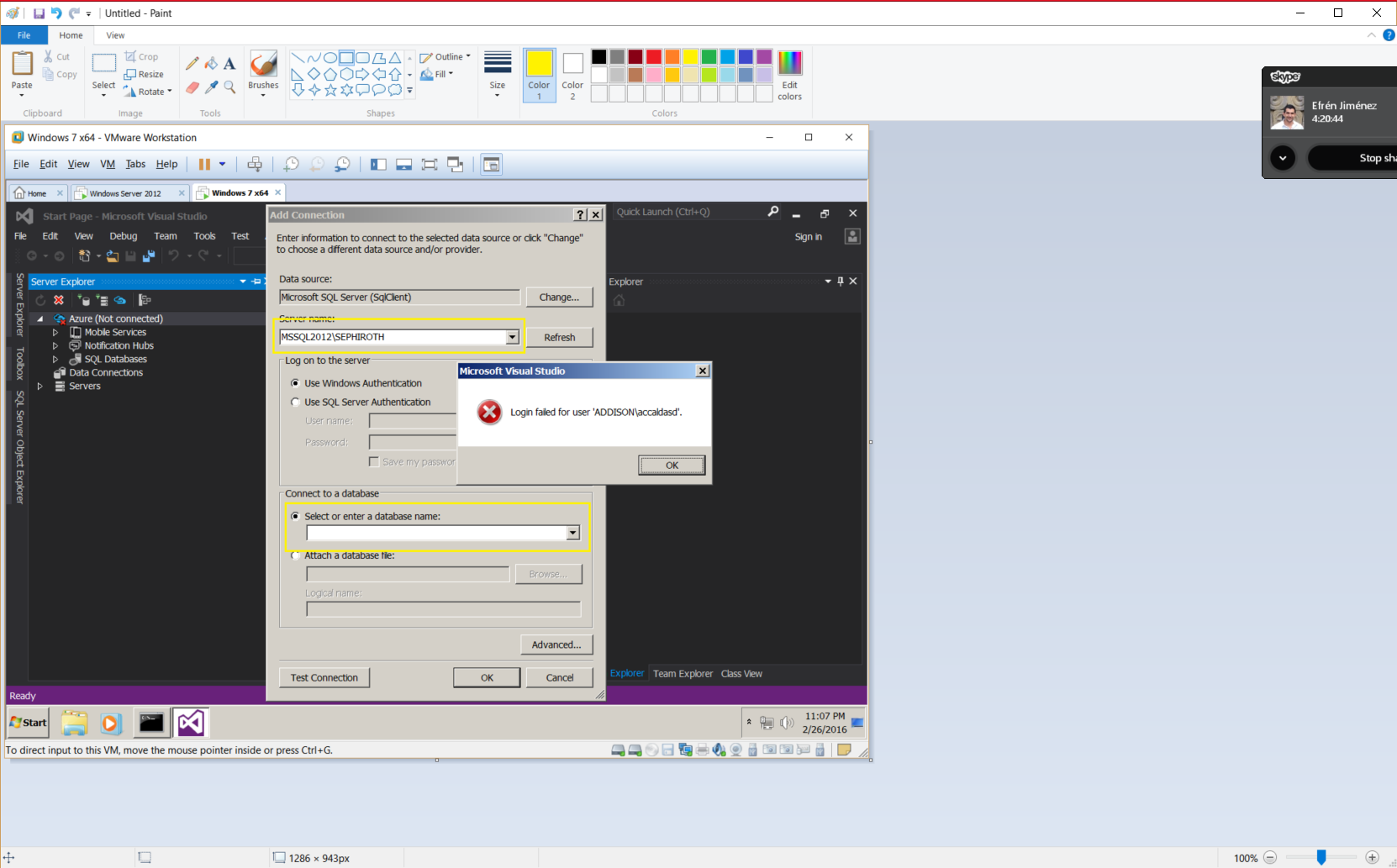


El proceso de autorización de cuentas de usuario basada en el servidor de dominio de activo activo, conlleva los siguientes pasos:

1. Verificar que el usuario no es capaz de iniciar una sesión autenticada con el SQL Server

Procedemos a iniciar una sesión con el nuevo usuario en una estación de trabajo (equipo terminal). El siguiente pantallazo muestra que se está utilizando el usuario creado en el dominio.





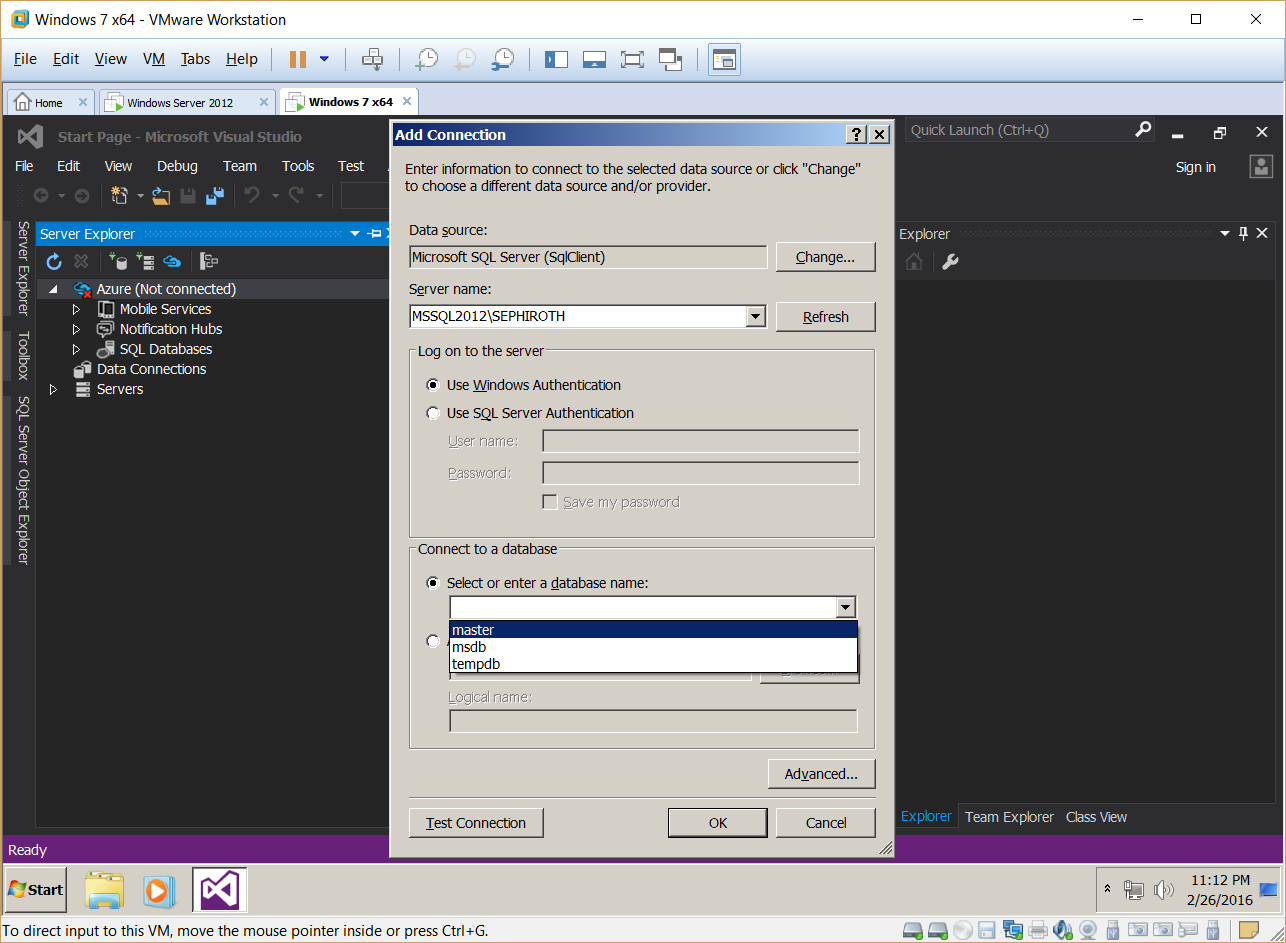
1. Crear un LOGIN asociado a la cuenta de dominio.

CREATE LOGIN [domain\user] FROM WINDOWS;

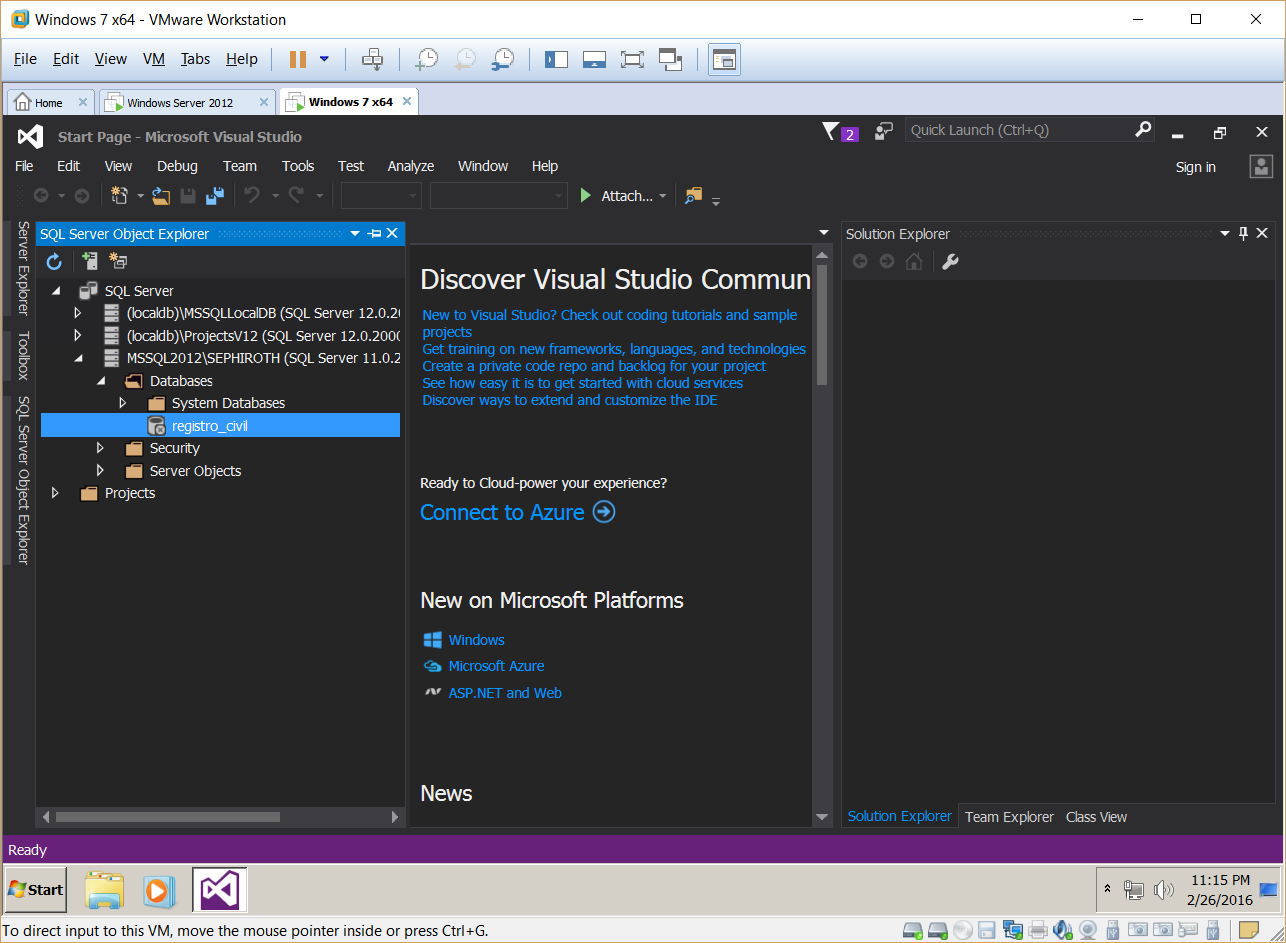
Observe que al crearse el LOGIN, el servidor permite al usuario ver la lista de base de datos creadas for default.

Ejemplo:

CREATE LOGIN [addison\accaldasd] FROM WINDOWS;



Sin embargo, si hacemos uso de la herramienta de exploración de objetos, notaremos que no es posible establecer una conexión con el schema registro\_civil, el cual ya está configurado en el sistema y posee una tabla. Observe como sobre el icono del schema registro\_civil, está superpuesta una (x), que señala que no es posible establecer la conexión.

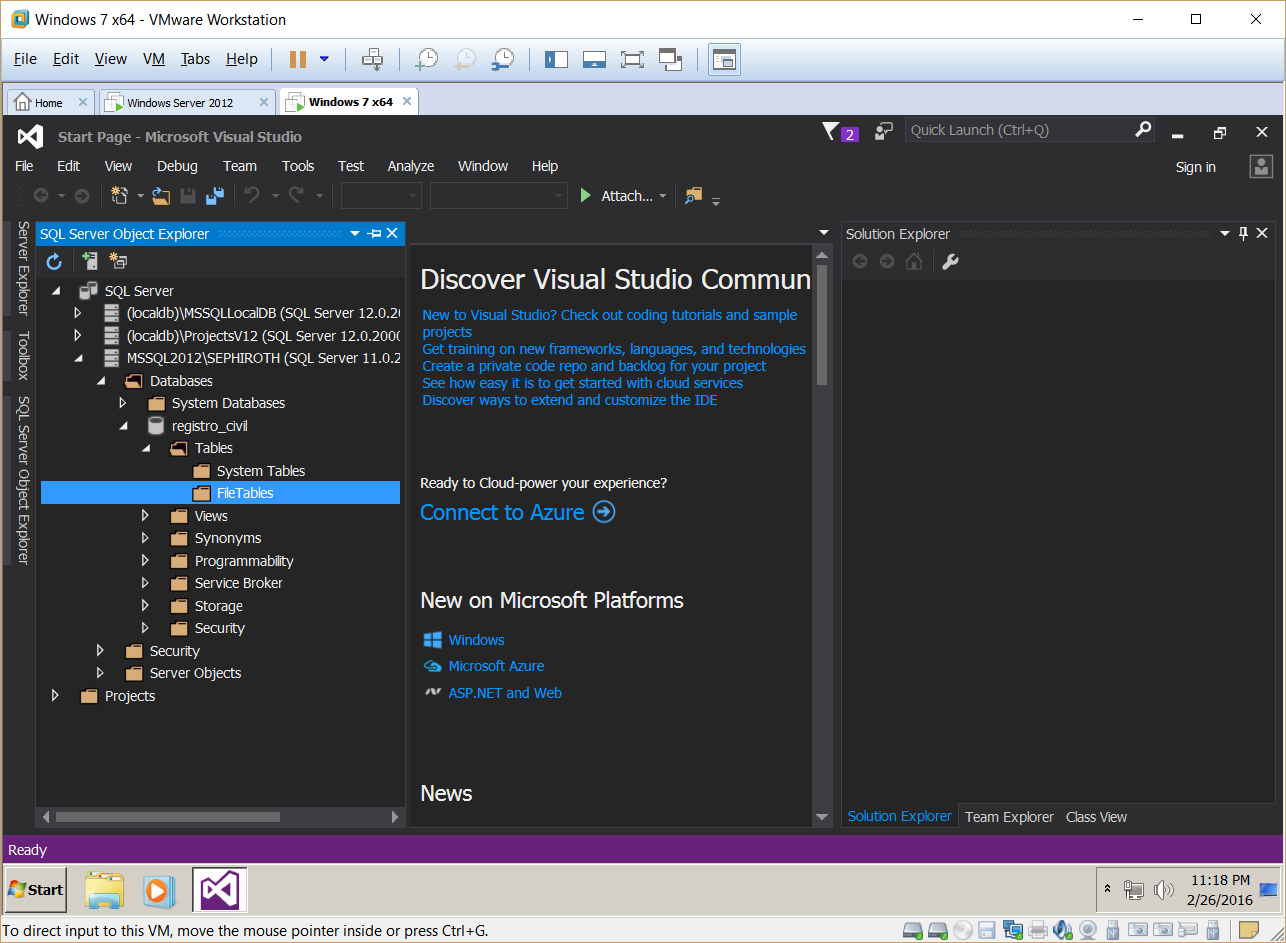


3. Crear un USER asociado al login

CREATE USER [domain\user] FROM LOGIN [domain\user];

Ejemplo:

CREATE USER [addison\accaldasd] FROM LOGIN [addison\accaldasd];



Con la configuración anterior se ha asociado un LOGIN a una cuenta de USER en el servidor el SQL Server. En esta ocasión, el cliente puede establecer la conexión con el schema, sin embargo, no le es posible ver los objectos o tablas configurados en el schema.

4. Habilitar el acceso del usuario de dominio al servidor de base de datos.

(<https://msdn.microsoft.com/en-us/library/ms173463.aspx>)

GRANT CONNECT TO [domain\user];

Ejemplo:

**Nota**: en versiones anteriores de SQL server, se utilizaba el stored procedure sp\_grantdbaccess para habilitar el permiso de conexión a la base de datos. Este procedimiento se considera obsoleto y e recomienda no utilizarse ya que va a ser removido en futuras versiones de SQL Server. (<https://msdn.microsoft.com/en-us/library/ms178013.aspx>)

EXEC sp\_grantdbaccess [domain\user], [domain\user]

Observe que, de esta manera, el usuario puede establecer una sesión con el servidor de SQL Server, no obstante, podrá notar que este no puede más que establecer una sesión, sin embargo, no puede acceder a las operaciones comunes de la base de datos

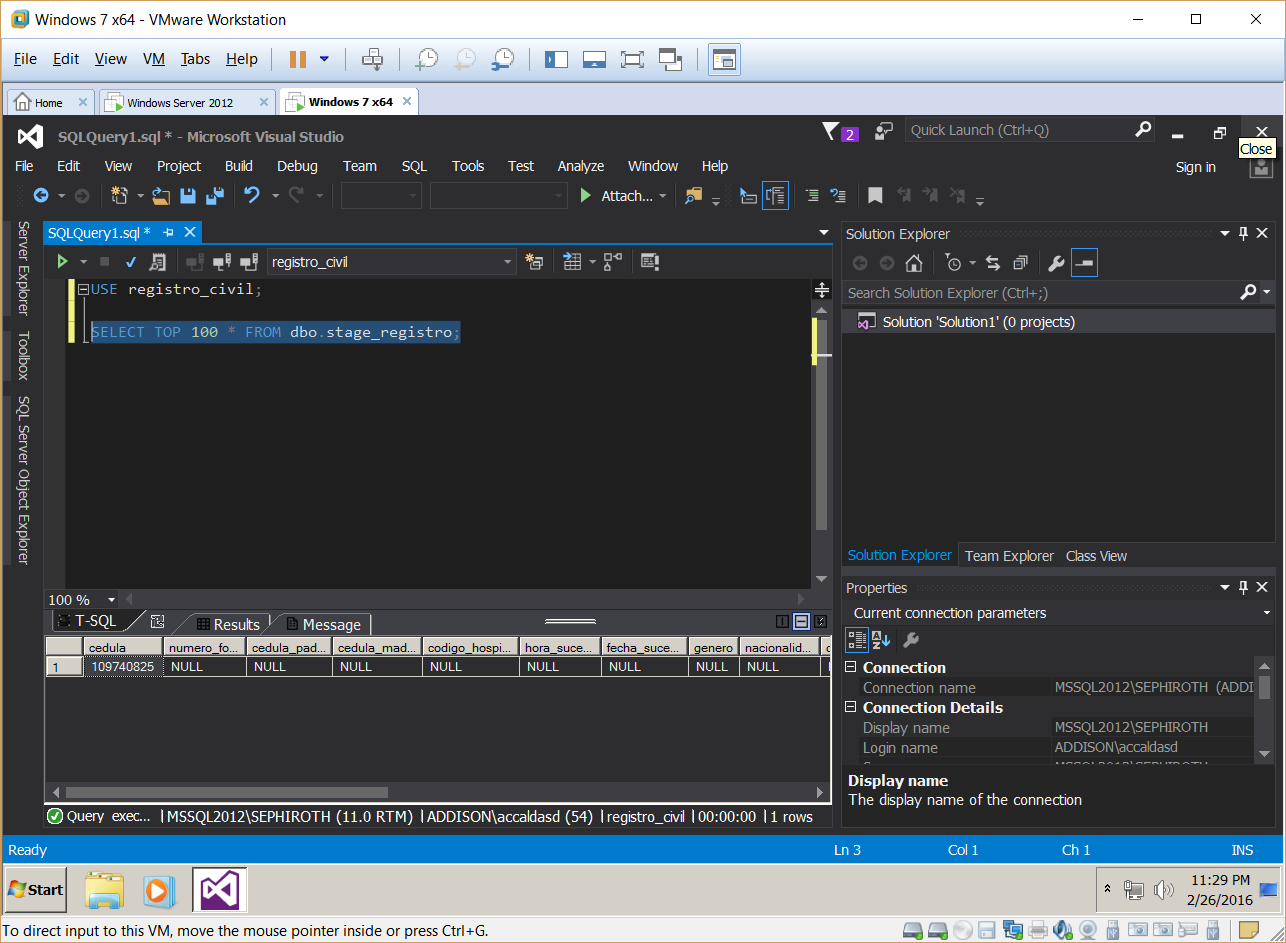
5. Habilitar los permisions apropiados en la tabla.

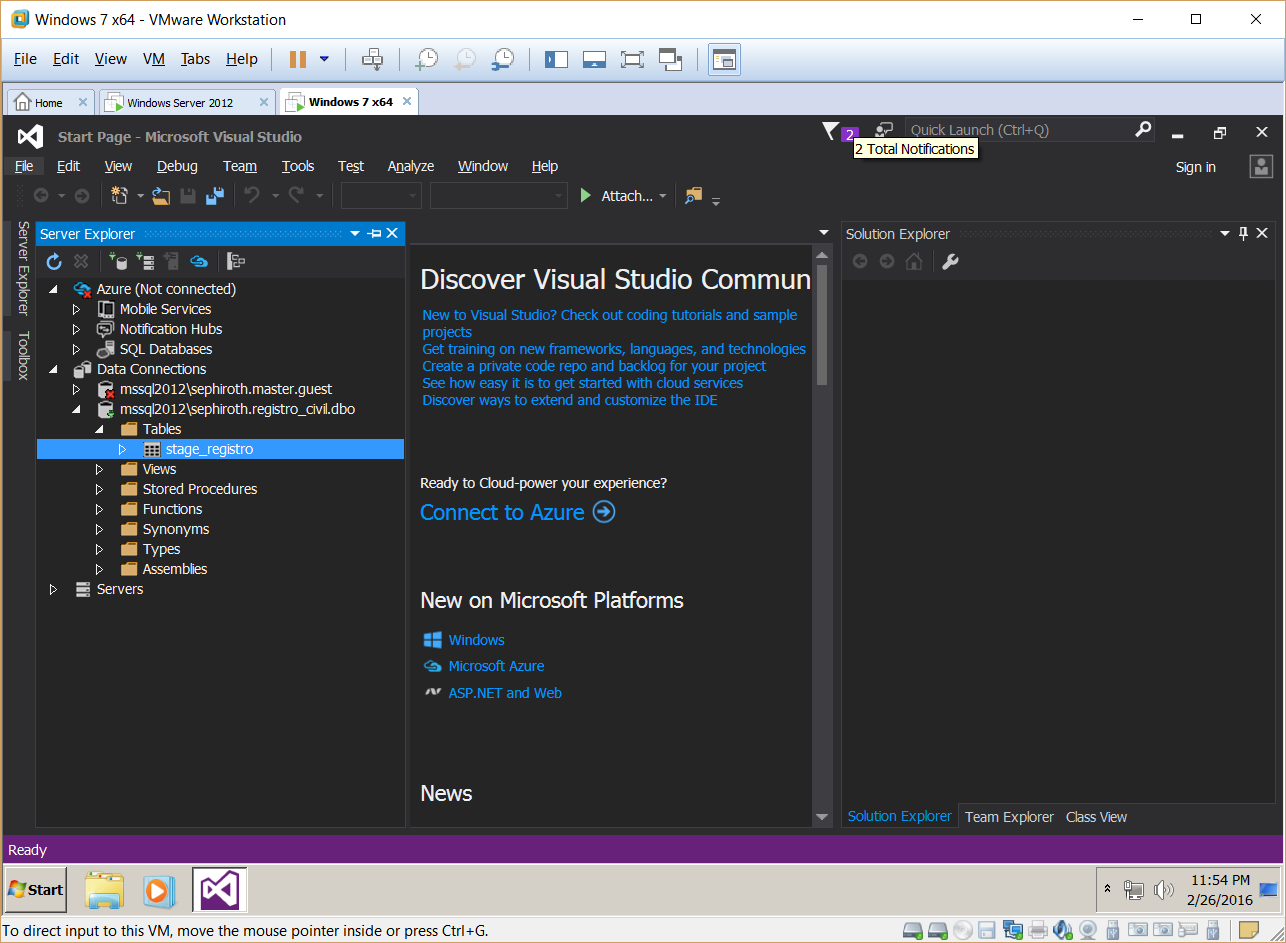
GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.Tablename TO [domain\user];

Ejemplo:

GRANT SELECT, INSERT, UPDATE, DELETE ON dbo.stage\_registro TO [addison\accaldasd];

De esta manera, ya es posible ejecutar un comando de selección sobre la table. Observe, sin embargo, que el browser de objetos, no es posible listar las tablas, debido a que estos es un permiso separada.





Reporte permisos de usuarios (servidor)

SELECT

a.name as Name

, a.type\_desc AS LoginType

, a.default\_database\_name AS DefaultDBName

, ISNULL(SUSER\_NAME(b.role\_principal\_id),'public') AS AssociatedServerRole

, c.class\_desc AS ClassDesc,c.permission\_name AS ServerLevelPermission

, c.state\_desc AS PermissionState

FROM sys.server\_principals a

LEFT JOIN sys.server\_role\_members b

ON a.principal\_id=b.member\_principal\_id

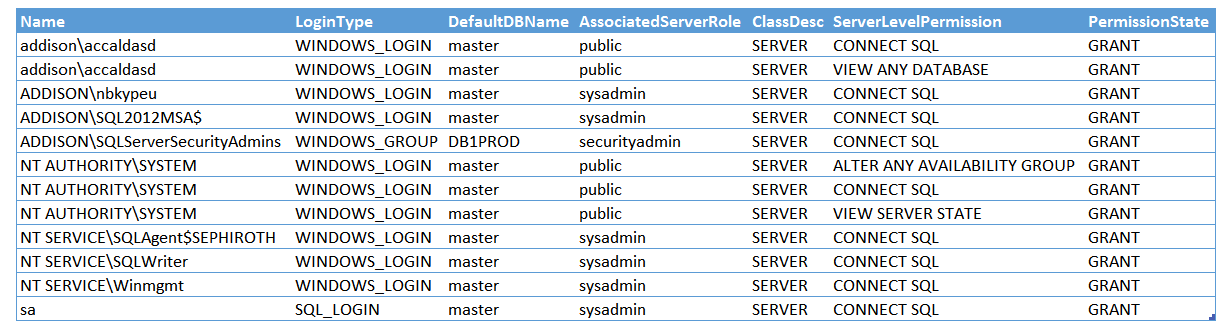
JOIN sys.server\_permissions c

ON a.principal\_id = c.grantee\_principal\_id

WHERE LEFT(a.name,2) <> '##'

AND a.name <> 'public'

ORDER BY Name, LoginType;



--Reporte permisos de usuarios (base de datos):

SELECT

pr.name AS Name

,pr.type\_desc AS LoginType

, pr.authentication\_type\_desc AS AuthType

, pe.class\_desc AS ClassDesc,pe.permission\_name AS DatabaseLevelPermission

,pe.state\_desc AS PermissionState

FROM sys.database\_permissions pe

JOIN sys.database\_principals pr

ON pr.principal\_id = pe.grantee\_principal\_id

WHERE pr.name <> 'public'

AND LEFT(pr.Name,2) <> '##'

ORDER BY Name, LoginType;

