TEC | Tecnológico de Costa Rica

CURSO

Bases De Datos 1 GR 01

Profesor

Rodrigo Nuñez Nuñez

Alumnos

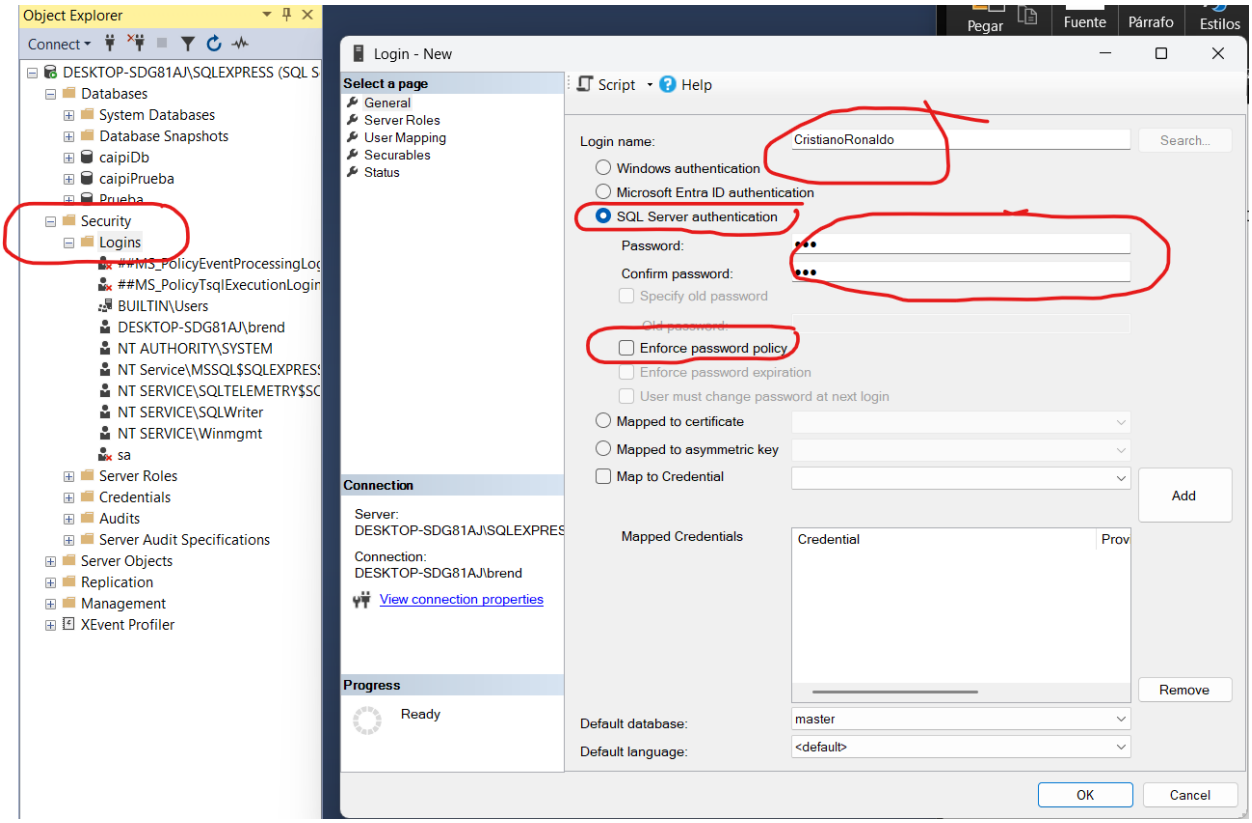
Brendan Ramírez Campos

Andrés Baldi Mora

Geovanni Esquivel Cortes

Victor Andrés Fung Chiong

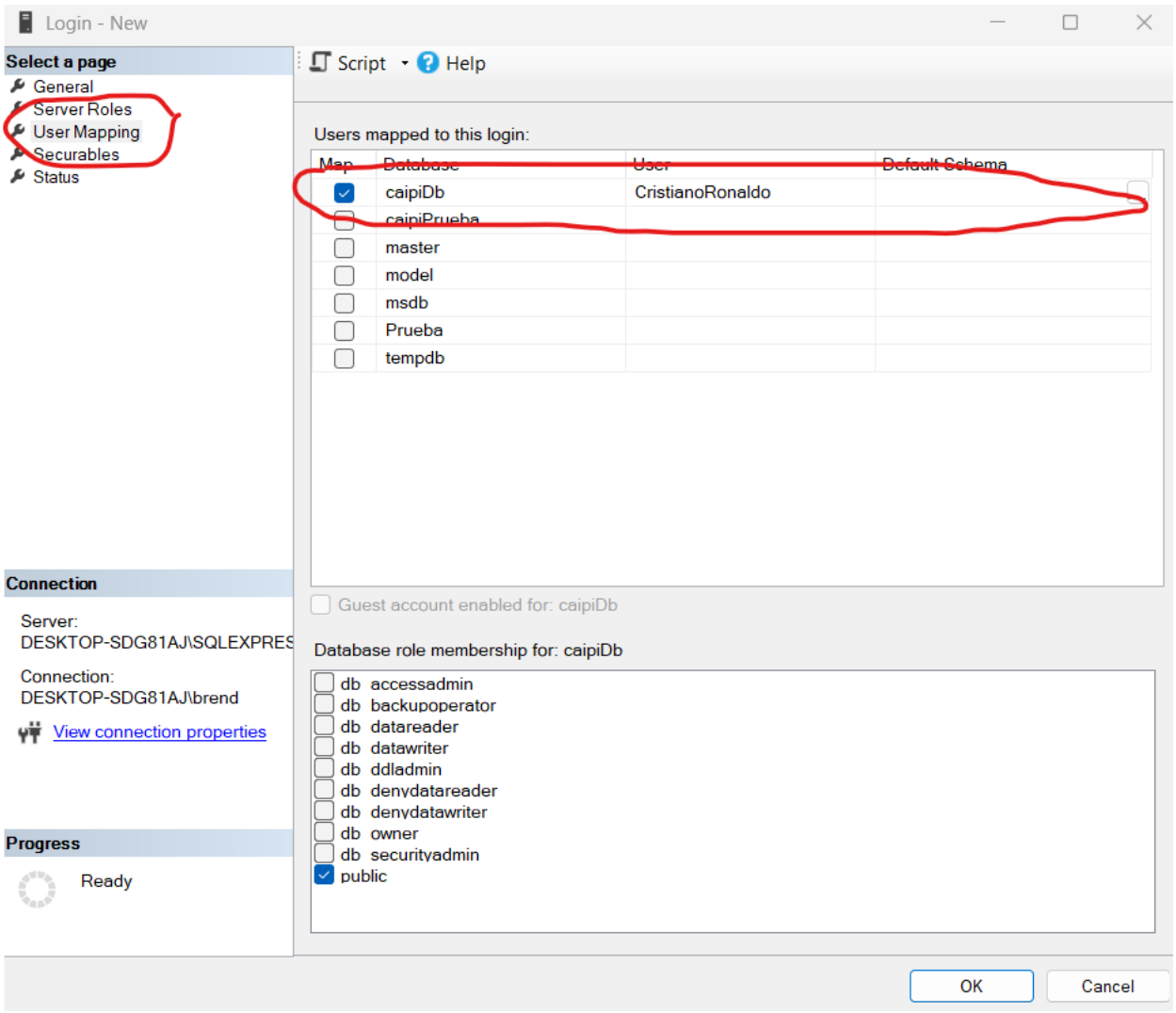
2025



Primero vamos a extender del lado izquierdo Security, seguido de eso click derecho en Logins y le damos a “New Login”, se nos va a abrir esa ventana donde vamos a poder ingresar el “Login name”, preferiblemente se selecciona “SQL Server Authentication”, ingresamos una contraseña y si el usuario es únicamente para testeos desactivamos la opción “Enforce password policy”

Usuario a registrar:

Usuario 1 CristianoRonaldo, Contra CR7

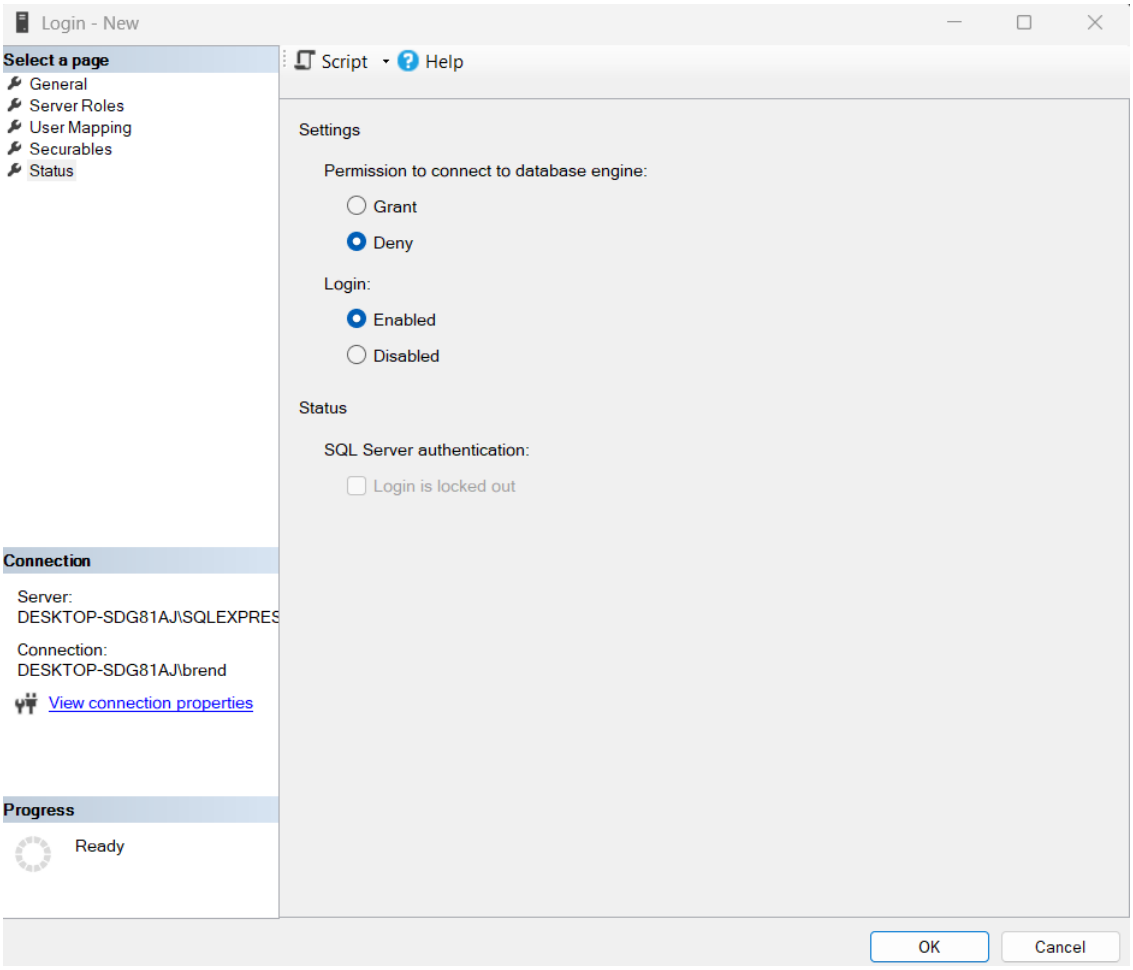


Si se quiere asociar solo a Bases de datos especificas selecciónelas en el apartado específico, el usuario será automáticamente creado en la base que se haya seleccionado.

Si se desea negar cualquier acceso no seleccione ninguna base o vaya a “Status” y seleccione “Deny “ para el login

Segundo usuario a agregar:

User 2 LoginDeny, contra 1234



Metodo de T-SQL:

-- crear login con acceso

CREATE LOGIN CristianoRonaldo WITH PASSWORD = CR7;

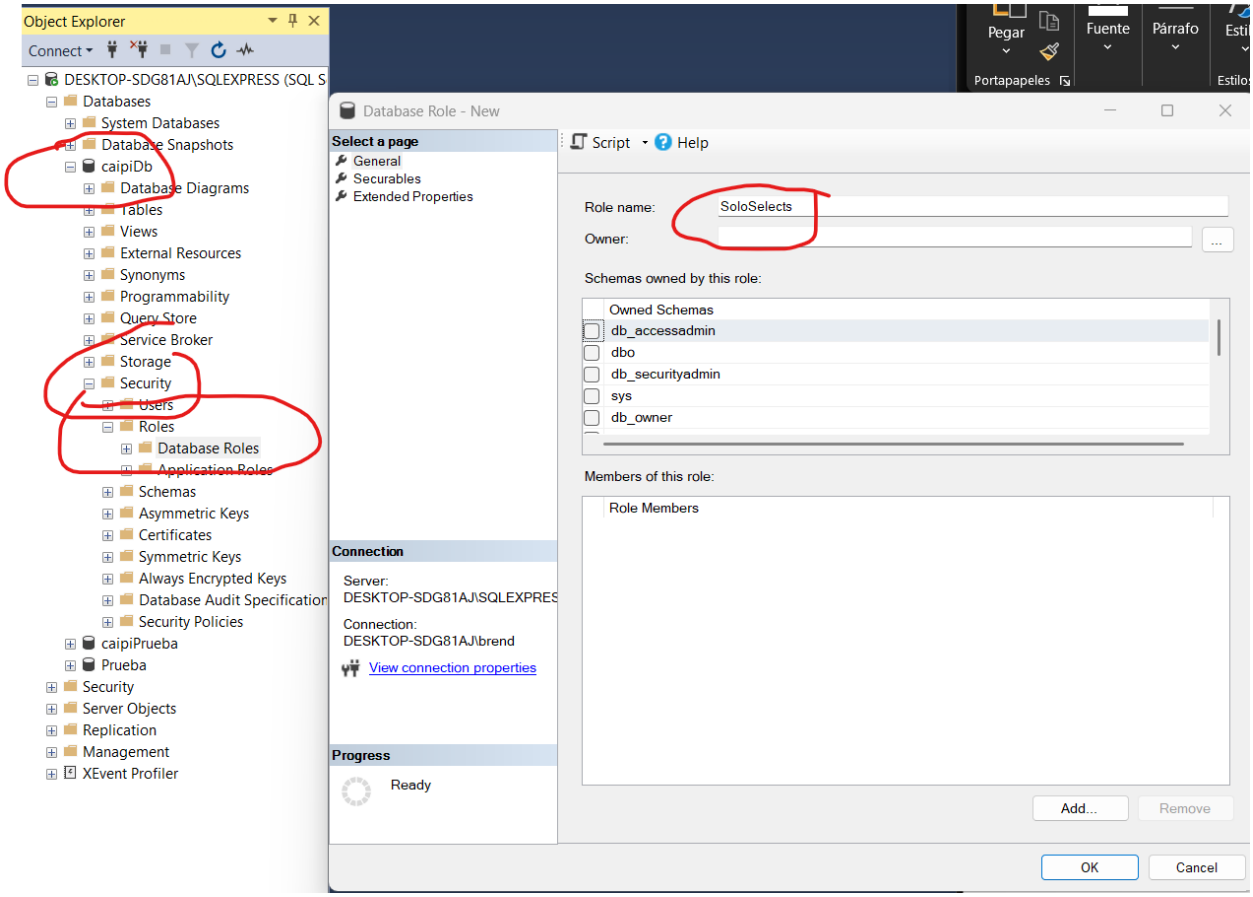
USE caipiDb;

CREATE USER CristianoRonaldo FOR LOGIN CristianoRonaldo;

-- crear login acceso denegado

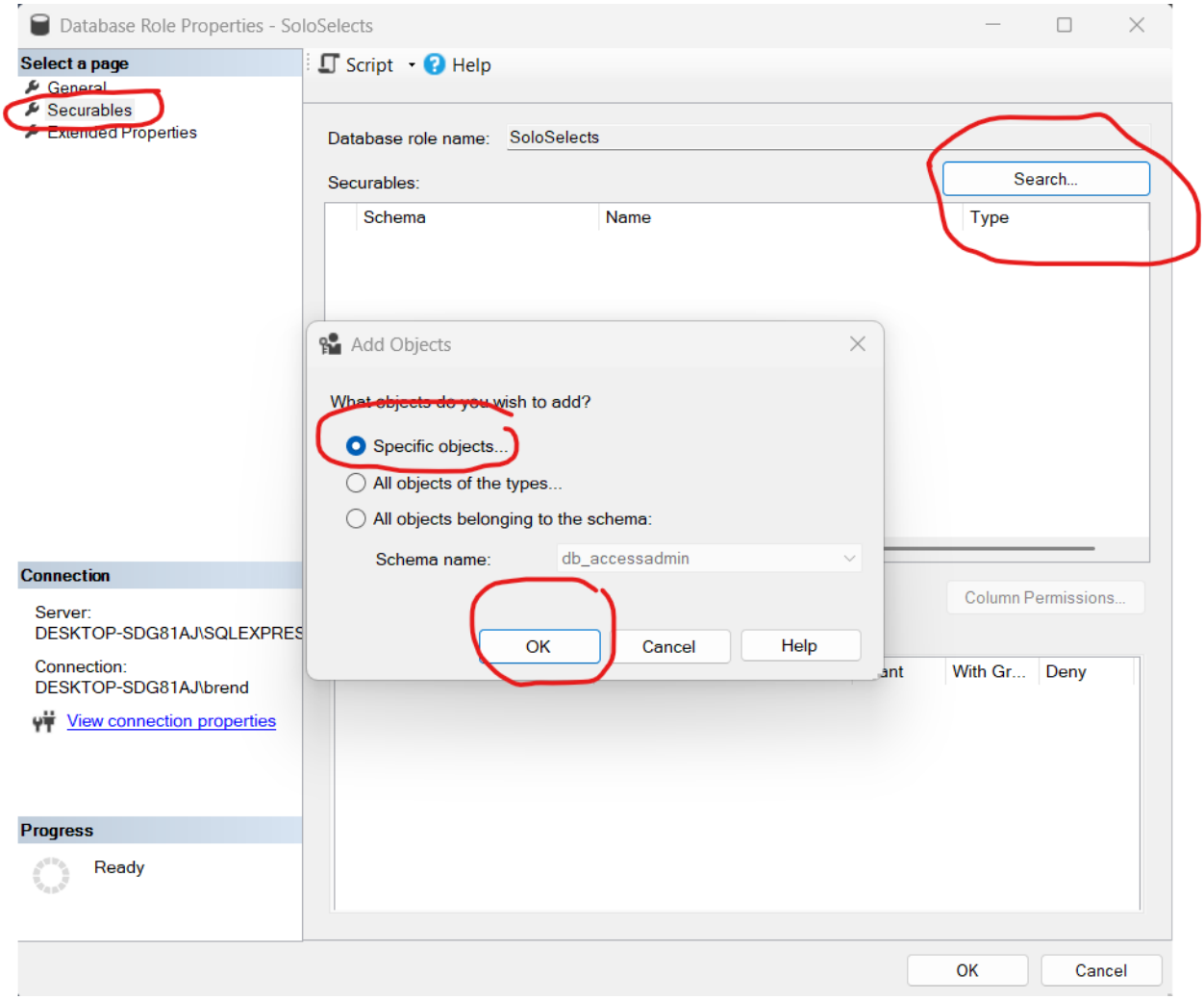
CREATE LOGIN LoginDeny WITH PASSWORD = 123;

--No hay mapping = No hay acceso a la base

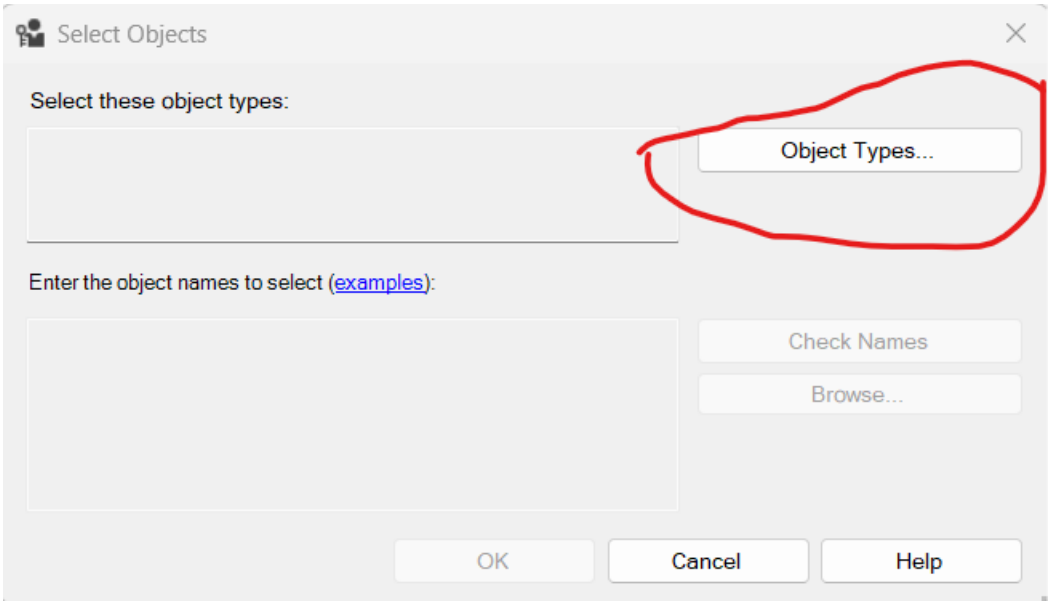


Para crear un role en el que solo se permite realizar Selects

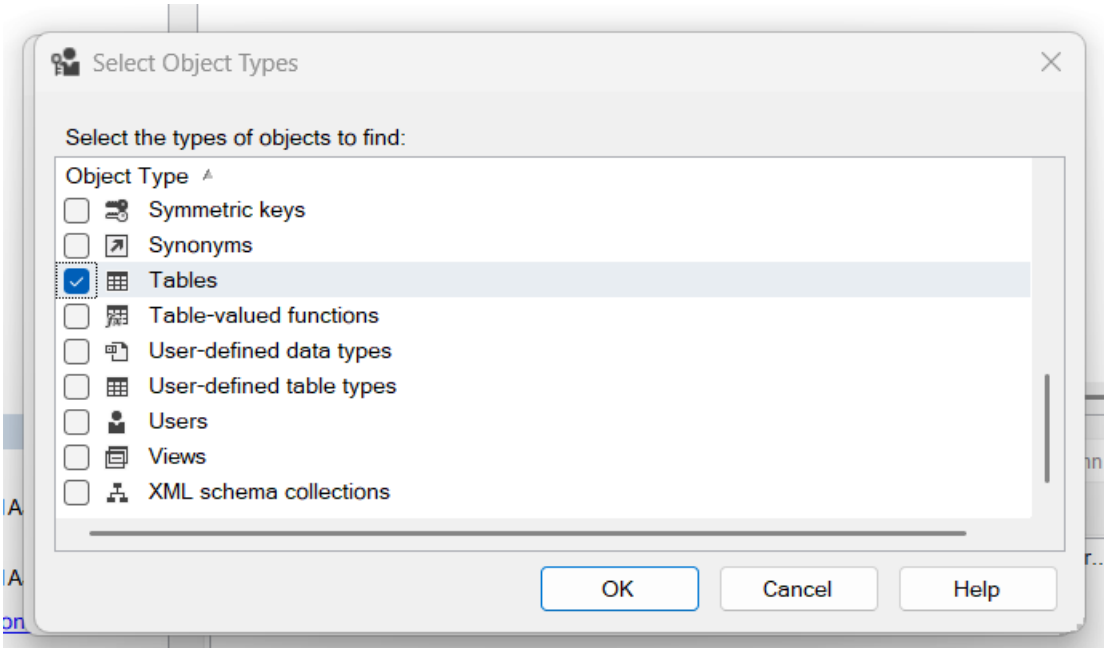
Primero extienda la database en la que se desea crea el rol, extienda “Security” dentro de la database, luego extienda “Roles”, y en “Database Roles” de click derecho y seleccione “New Database Role”, aparece esa ventana, ingrese un nombre.



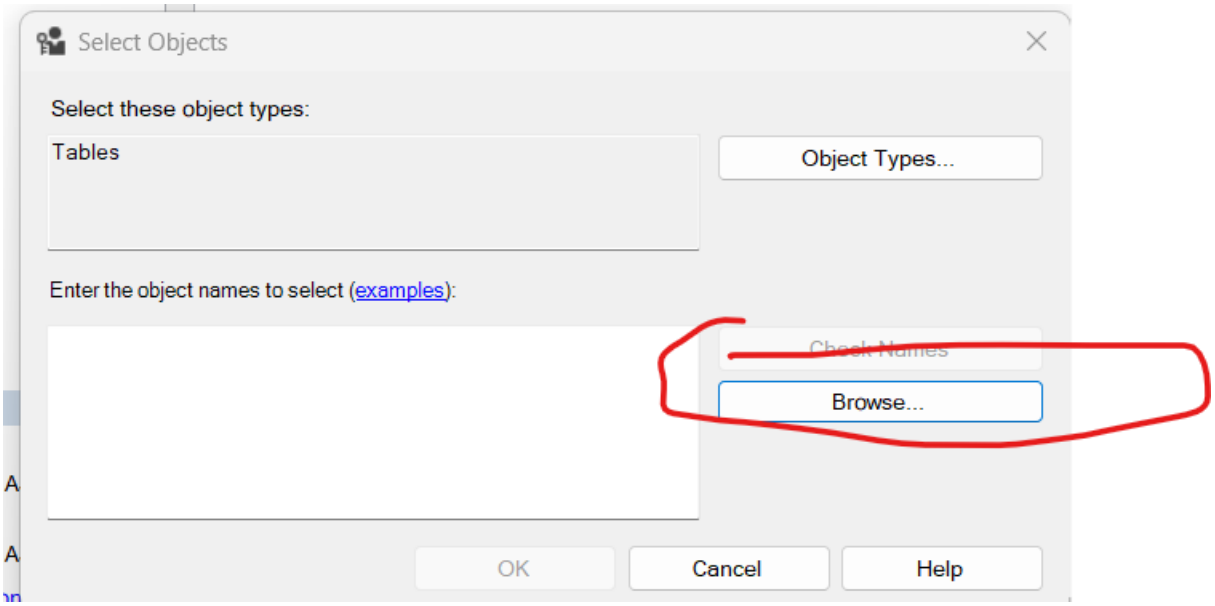
Luego en “Securables”, click Search, aparece esa ventana y seleccione “Specific Objects” de ok y se muestra lo siguiente



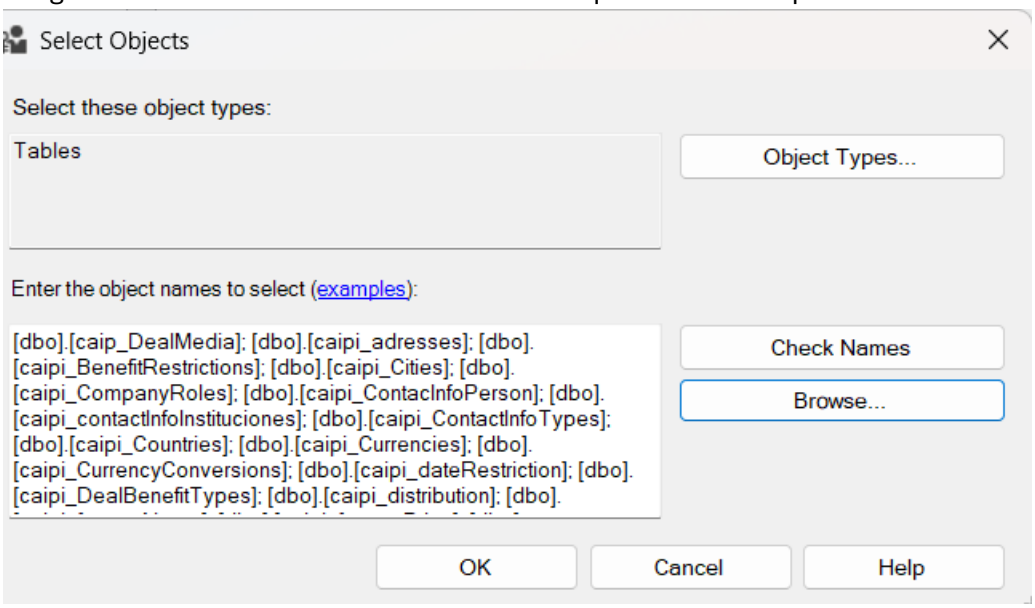
Seleccione “Object Types”



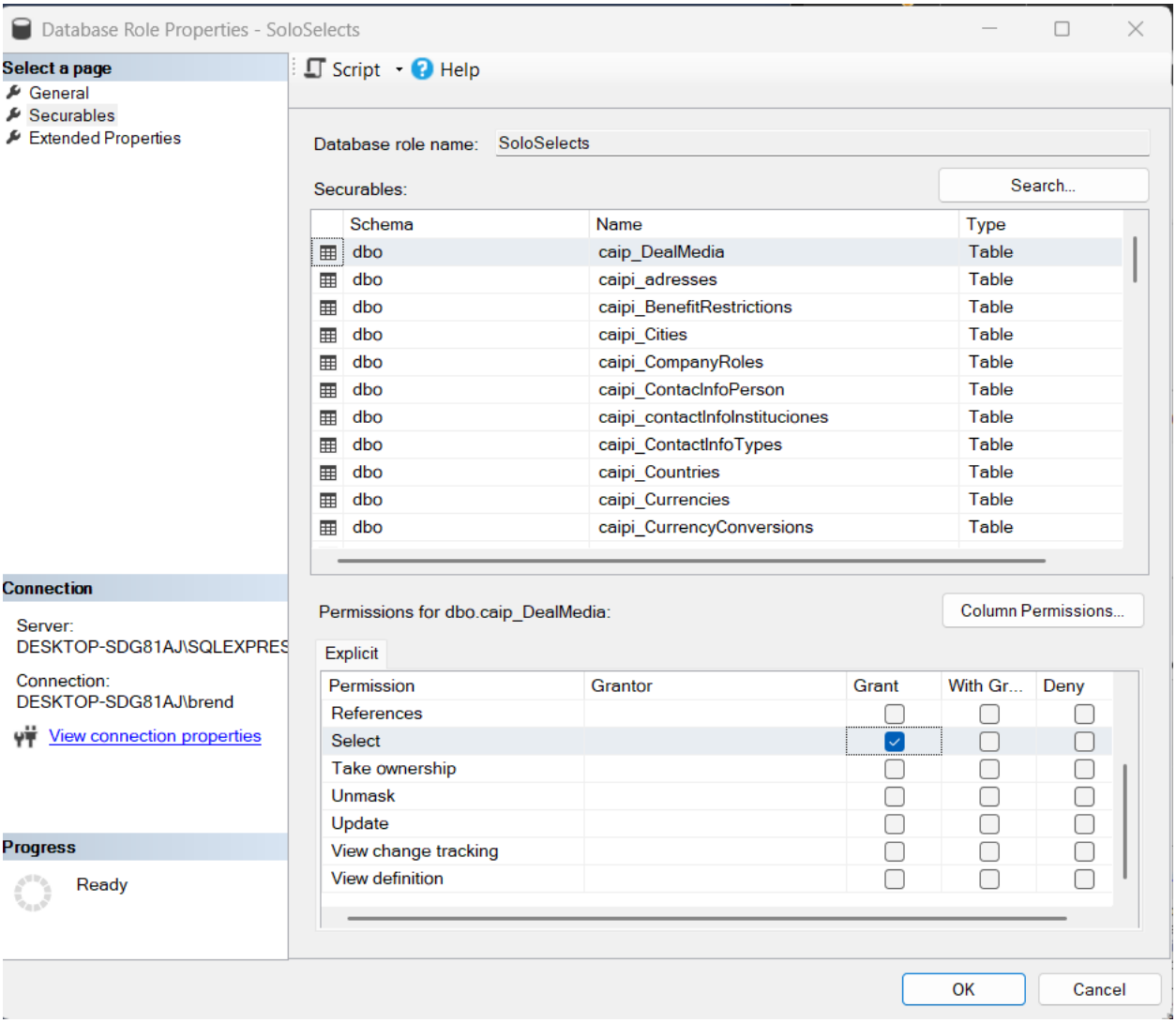
Marque “Tables



Luego en “Browse” seleccione las tablas a las que desea darle permiso

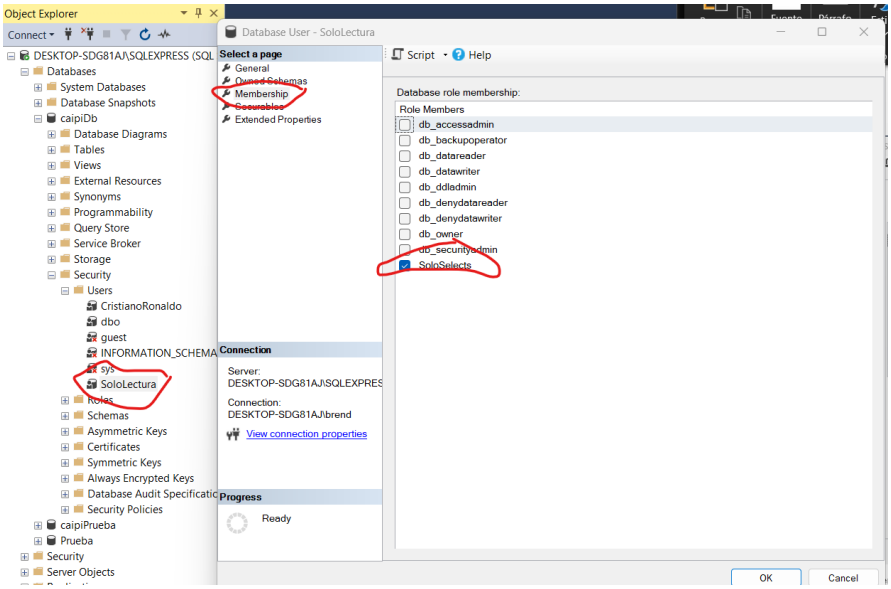


Se va a ver de esta manera



Una vez se le de ok, baje en los permisos y seleccione el que se desea garantizar, en mi caso únicamente le voy a dar permisos para realizar selects

Login/User 3 Name: SoloLectura, Contra: en blanco



Para asignarle el rol al usuario dele click derecho al usuario, luego propiedades y una vez ahí, diríjase a Membership, y seleccione el rol que se le desea agregar

Metodo T-SQL

```
-- Create role

USE YourDatabase;

CREATE ROLE SelectOnlyRole;


-- Grant SELECT on specific table

GRANT SELECT ON dbo.YourTable TO SelectOnlyRole;


-- Create users

CREATE LOGIN TestUser2 WITH PASSWORD = 'Password123';

CREATE USER TestUser2 FOR LOGIN TestUser2;

ALTER ROLE SelectOnlyRole ADD MEMBER TestUser2;


CREATE LOGIN TestUser3 WITH PASSWORD = 'Password123';

CREATE USER TestUser3 FOR LOGIN TestUser3;

-- No role assignment = no SELECT permission
```

Manejo de los SP’s

No se logró realizar de manera grafica voy a explicarlo usando T-SQL

```
-- Crear un procedimiento almacenado de ejemplo
CREATE PROCEDURE dbo.sp_GetPartnerDealInfo
AS
BEGIN
  SELECT p.partnerDealId, p.dealDescription, b.name AS BenefitType
  FROM dbo.caipi_PartnerDeals p
  JOIN dbo.caipi_DealBenefitTypes b ON p.partnerDealId = b.dealBenefitTypesId
  WHERE p.isActive = 1;
END;
GO
-- Crear un rol y asignar permisos (al no poner nada, no se le otorga ningún permiso)
CREATE ROLE SPExecutorRole;
GO

-- Denegar acceso a las tablas
DENY SELECT ON dbo.caipi_PartnerDeals TO SPExecutorRole;
DENY SELECT ON dbo.caipi_DealBenefitTypes TO SPExecutorRole;
GO

-- Permitir ejecución del SP
GRANT EXECUTE ON dbo.sp_GetPartnerDealInfo TO SPExecutorRole;
GO

-- Asignar un usuario al rol (opcional)
CREATE USER EjemploUsuario WITHOUT LOGIN;
ALTER ROLE SPExecutorRole ADD MEMBER EjemploUsuario;
GO
```

Esta configuración permite que los usuarios en SPExecutorRole ejecuten el stored procedure pero no puedan consultar directamente las tablas.

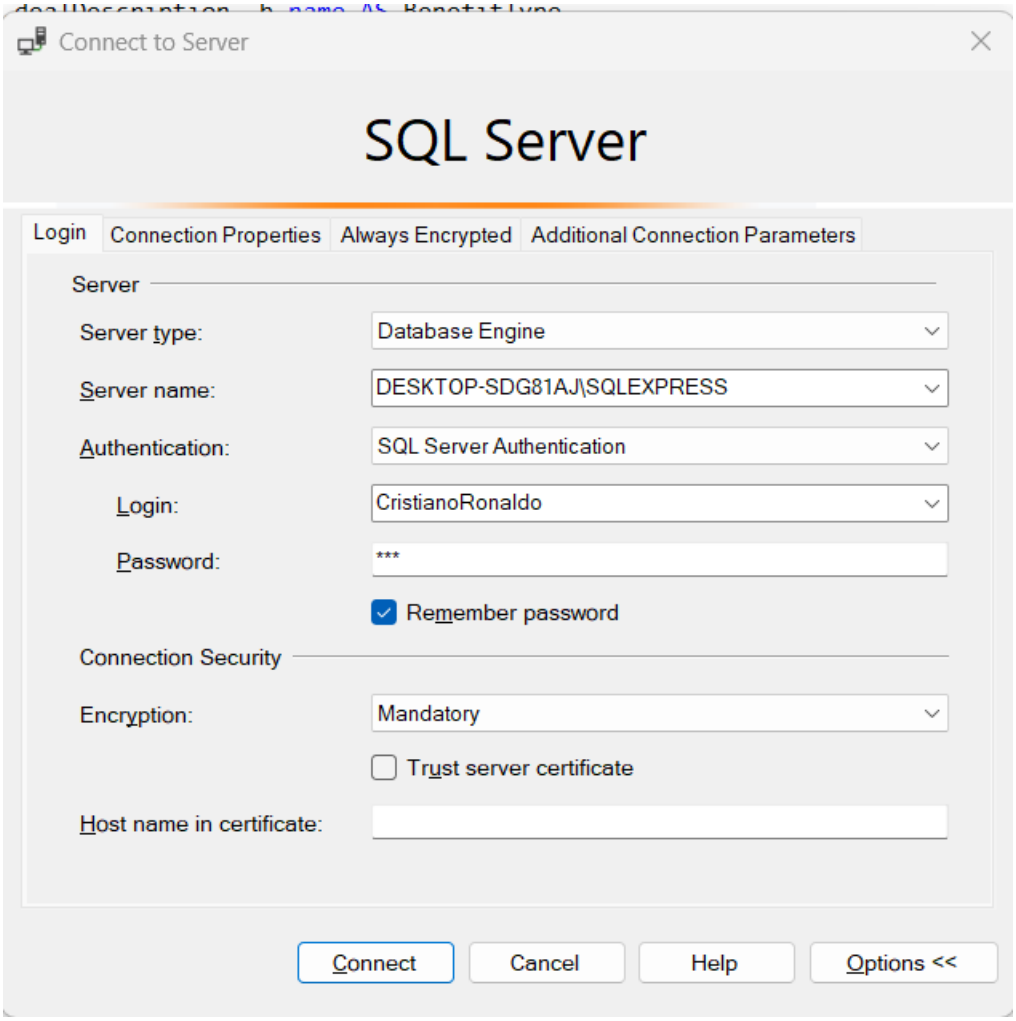
En respuesta a la pregunta

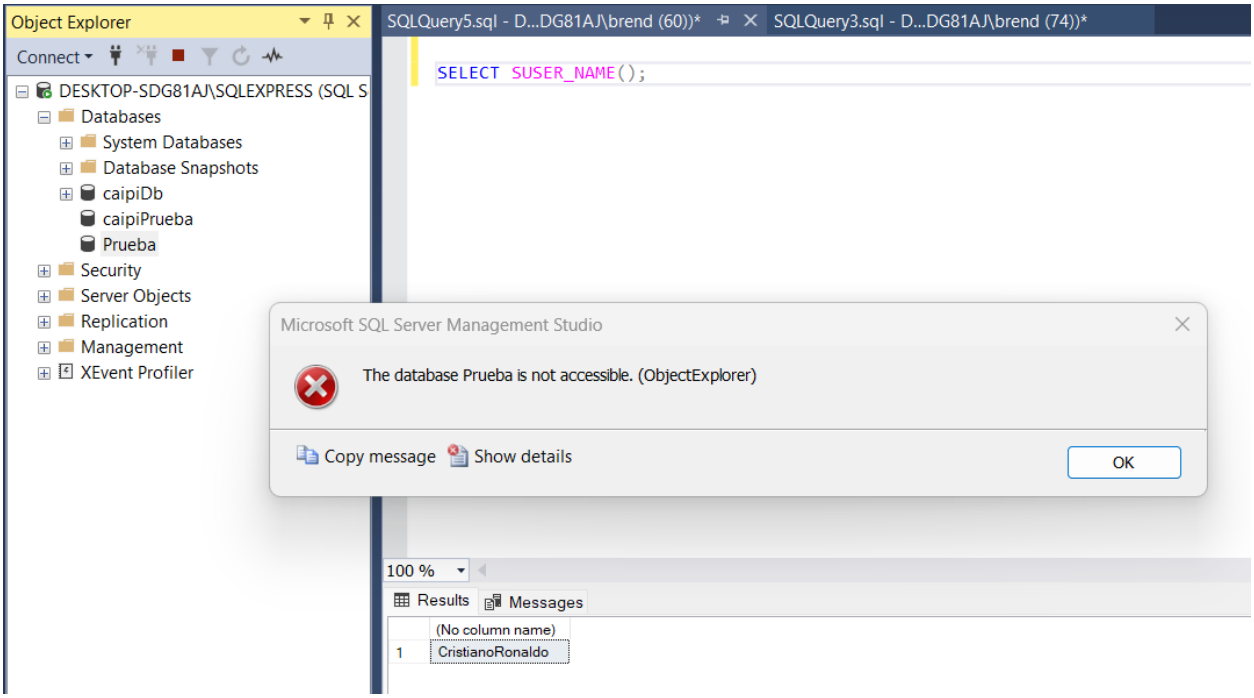
Sí, puedes ejecutar el stored procedure aunque tengas las tablas restringidas, siempre que:

1. El stored procedure tenga los permisos necesarios (generalmente a través de la propiedad EXECUTE AS OWNER o con un usuario con permisos suficientes)
2. Solo se le haya concedido permiso EXECUTE al usuario/rol
3. No se le hayan concedido permisos directos sobre las tablas

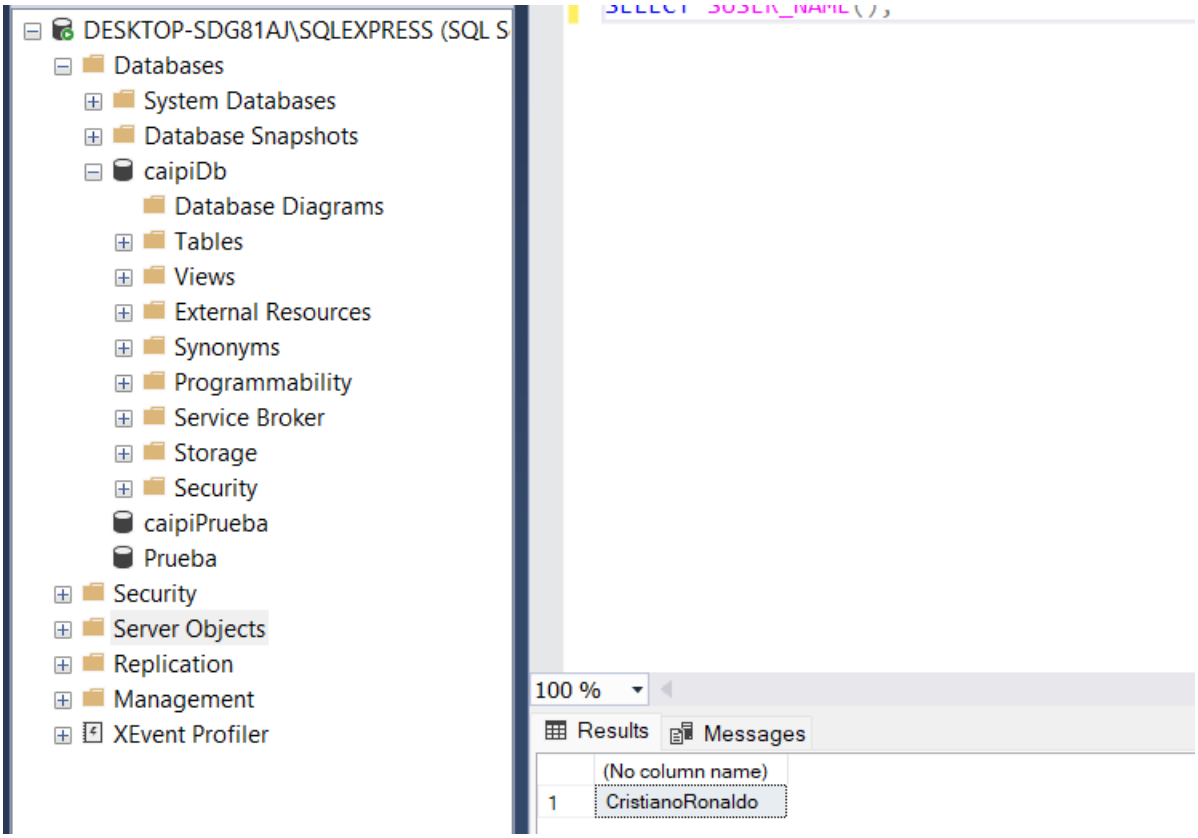
Muestreo del funcionamiento

Ingreso con CristianoRonaldo

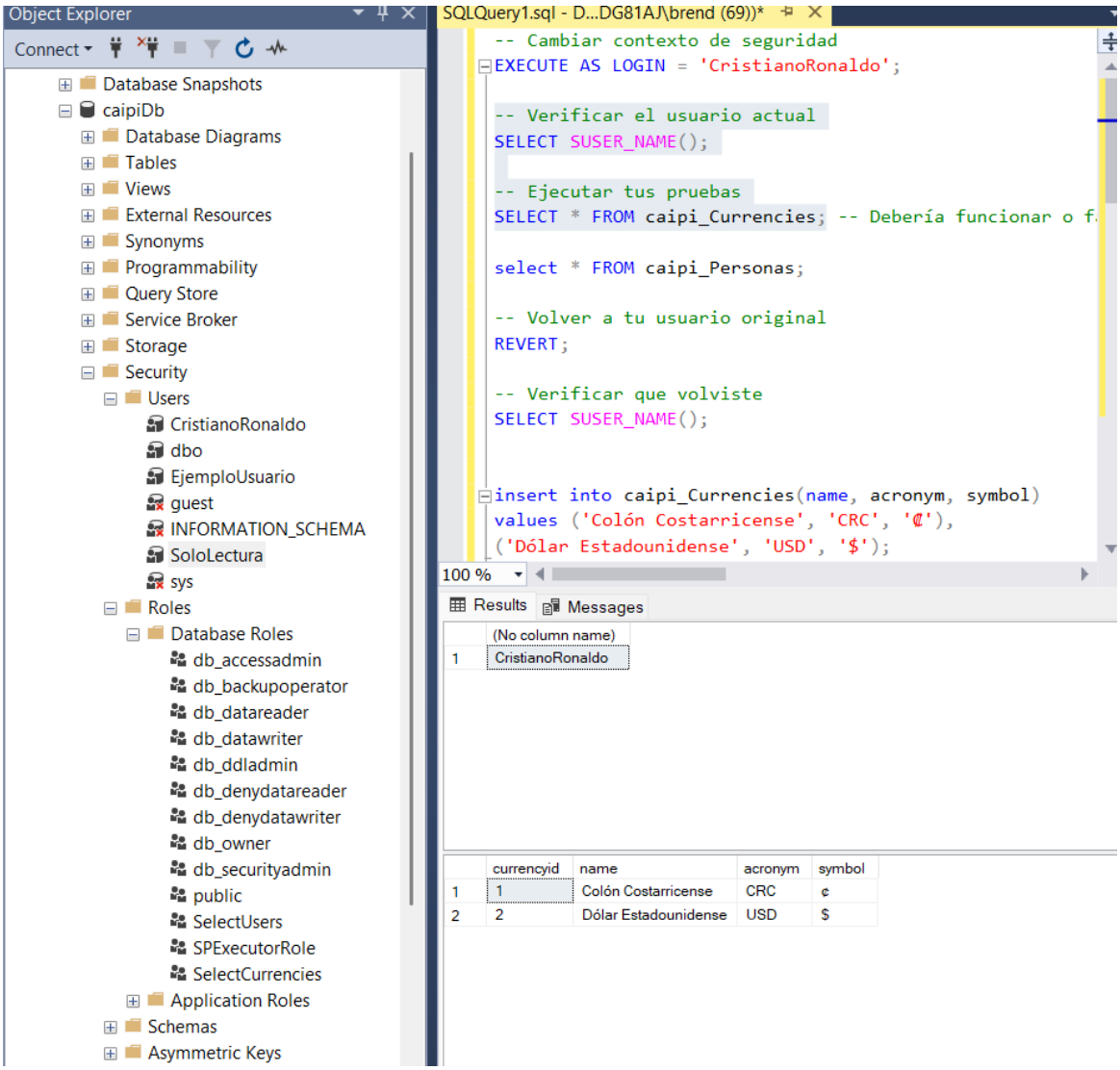




Se da un muestreo en el que no se tiene acceso a las bases a las cuales no se le dio permiso al user



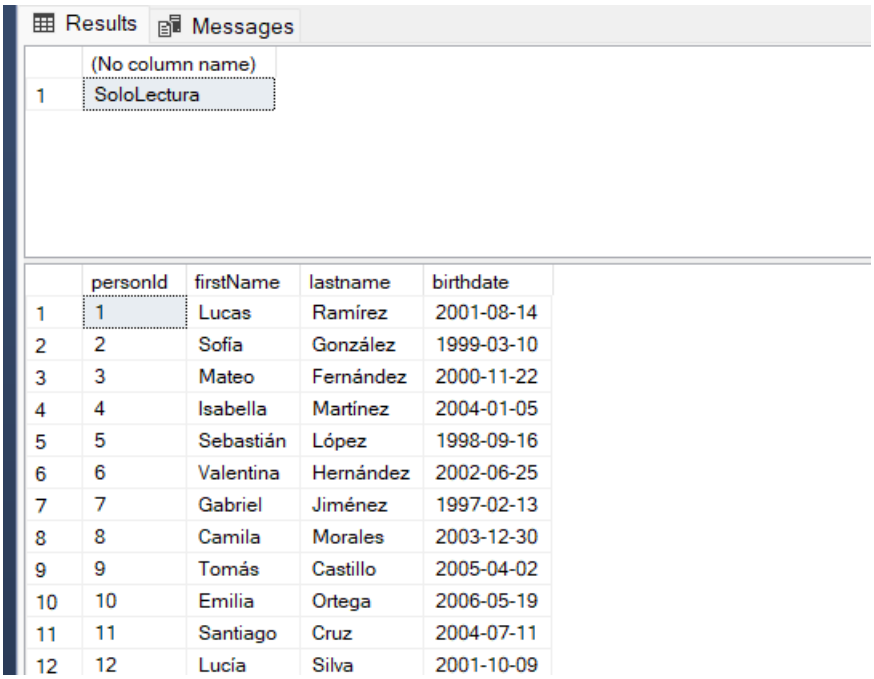
Se muestra que en caipiDb si tiene permisos



Cuando se realiza el select en “Caipi_Currencies” se puede realizar, pero cuando es en otra tabla no, esto con el usuario de CristianoRonaldo

```
Msg 229, Level 14, State 5, Line 8
The SELECT permission was denied on the object 'caipi_adresses', database 'caipiDb', schema 'dbo'.
```

Por ejemplo con este otro usuario “SoloLectura” no puede hacer select en “Caipi_ Currencies” pero si en “Caipi_Personas”

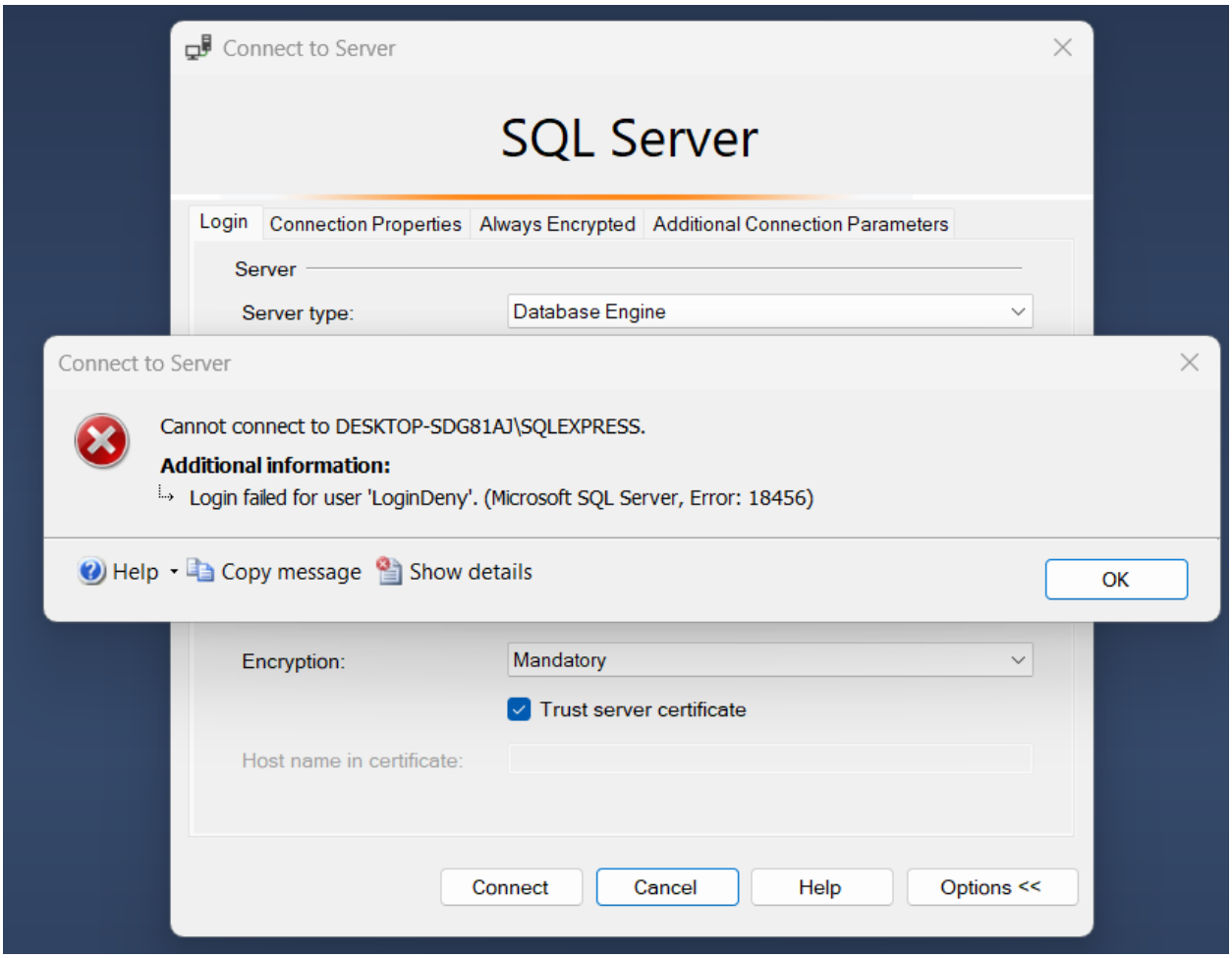


Results Messages

	(No column name)
1	SoloLectura

	personId	firstName	lastname	birthdate
1	1	Lucas	Ramírez	2001-08-14
2	2	Sofía	González	1999-03-10
3	3	Mateo	Fernández	2000-11-22
4	4	Isabella	Martínez	2004-01-05
5	5	Sebastián	López	1998-09-16
6	6	Valentina	Hernández	2002-06-25
7	7	Gabriel	Jiménez	1997-02-13
8	8	Camila	Morales	2003-12-30
9	9	Tomás	Castillo	2005-04-02
10	10	Emilia	Ortega	2006-05-19
11	11	Santiago	Cruz	2004-07-11
12	12	Lucía	Silva	2001-10-09

Entonces en respuesta a lo propuesto, si es posible que existan roles que puedan hacer select a una tabla y otros que no, pero si puedan hacerlo a otra tabla



Aquí se da un muestreo de como funciona cuando se le prohíbe conectarse a un Login

RLS
No se puede realizar por medio de la parte grafica, entonces se va a hacer uso de T-SQL

```
USE [caipiDb]

GO


-- Crear esquema para seguridad si no existe

IF NOT EXISTS (SELECT * FROM sys.schemas WHERE name = 'Security')

BEGIN

   EXEC('CREATE SCHEMA Security')

END

GO
```

```
-- Crear función de predicado
CREATE FUNCTION Security.fn_securitypredicate(@userId int)
RETURNS TABLE
WITH SCHEMABINDING
AS
RETURN SELECT 1 AS fn_securitypredicate_result
WHERE @userId = USER_ID() OR USER_NAME() = 'dbo';
GO

-- Crear política de seguridad
CREATE SECURITY POLICY Security.userFilterPolicy
ADD FILTER PREDICATE Security.fn_securitypredicate(userId)
ON dbo.caipi_Users;
GO
```

Para la creación del certificado y de la llave asimétrica no se pudo usar el apartado grafico, por ende se usa T-SQL

Para la ejecución de este primero se debe crear una clave maestra para la base de datos, o bueno, por lo menos en nuestro caso

```
-- Crear la clave maestra de la base de datos
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'UnaClaveMuySegura123!';
GO

-- Verificar la existencia de la clave maestra
SELECT * FROM sys.symmetric_keys WHERE name = '##MS_DatabaseMasterKey##';
GO

-- Crear certificado
```

```
CREATE CERTIFICATE Certificado_CAIPI
WITH SUBJECT = 'Certificado para cifrado CAIPI',
EXPIRY_DATE = '20300101';
GO

-- Crear clave asimétrica
CREATE ASYMMETRIC KEY ClaveAsimetrica_CAIPI
WITH ALGORITHM = RSA_2048;
GO
```

Creación de llave simétrica

```
-- Crear clave simétrica protegida por el certificado
CREATE SYMMETRIC KEY ClaveSimetrica_CAIPI
WITH ALGORITHM = AES_256
ENCRYPTION BY CERTIFICATE Certificado_CAIPI;
GO
```

Para el cifrado de datos sensibles se puede usar este T-SQL, se realiza un cifrado de el password de la tabla “caipi_Users”

```
-- Primero, necesitamos modificar la tabla para almacenar datos cifrados
-- (Nota: Esto es un ejemplo, en producción necesitarías un proceso de migración cuidadoso)

-- Paso 1: Agregar una nueva columna para el password cifrado
ALTER TABLE dbo.caipi_Users
ADD password_cifrado varbinary(8000) NULL;
GO

-- Paso 2: Cifrar los datos existentes
```

```
OPEN SYMMETRIC KEY ClaveSimetrica_CAIPI
DECRYPTION BY CERTIFICATE Certificado_CAIPI;

UPDATE dbo.caipi_Users
SET password_cifrado = ENCRYPTBYKEY(KEY_GUID('ClaveSimetrica_CAIPI'),
CONVERT(varchar(250), password));

CLOSE SYMMETRIC KEY ClaveSimetrica_CAIPI;
GO

-- Paso 3: Verificar que los datos se cifraron correctamente
OPEN SYMMETRIC KEY ClaveSimetrica_CAIPI
DECRYPTION BY CERTIFICATE Certificado_CAIPI;

SELECT
  userId,
  password AS password_original,
  password_cifrado,
  CONVERT(varbinary(250), DECRYPTBYKEY(password_cifrado)) AS password_descifrado
FROM dbo.caipi_Users;

CLOSE SYMMETRIC KEY ClaveSimetrica_CAIPI;
GO
```

Para el descifrado se hace uso de este Stored Procedure

```
CREATE PROCEDURE dbo.DescifrarPasswordUsuario
  @userId int
AS
BEGIN
  SET NOCOUNT ON;

  -- Abrir la clave simétrica
  OPEN SYMMETRIC KEY ClaveSimetrica_CAIPI
  DECRYPTION BY CERTIFICATE Certificado_CAIPI;

  -- Descifrar y mostrar el password
  SELECT
    u.userId,
    p.firstName,
    p.lastname,
    CONVERT(varchar(250), DECRYPTBYKEY(u.password_cifrado)) AS password_descifrado
  FROM dbo.caipi_Users u
  JOIN dbo.caipi_Personas p ON u.personId = p.personId
  WHERE u.userId = @userId;

  -- Cerrar la clave
  CLOSE SYMMETRIC KEY ClaveSimetrica_CAIPI;
END
GO
```