PRACTICA DE LA SEMANA 11

```sql
-- 1A. Crear el Login SQL. Usar CHECK_POLICY=ON y CHECK_EXPIRATION=ON
-- Reemplaza 'tu_password_segura' con una contraseña real y segura.
CREATE LOGIN login_sql_alumno
    WITH PASSWORD = 'tu_password_segura',
    CHECK_POLICY = ON,
    CHECK_EXPIRATION = ON;
GO
```

creacion de login windows simulado

```sql
CREATE LOGIN [VICTUSHP\elvis] FROM WINDOWS;
GO
```

2. Comprobar Políticas y Forzar Expiración (Demostración)

Esta sección demuestra que las políticas están activas y cómo forzar una expiración, lo cual es parte del requisito del proyecto.

```sql
SELECT name, is_policy_checked, is_expiration_checked
FROM sys.sql_logins
WHERE name = 'login_sql_alumno';
GO

-- El script original del proyecto ya lo hace al crearlo, pero esta es la forma de forzarlo después.
ALTER LOGIN login_sql_alumno WITH CHECK_EXPIRATION = OFF; -- Paso temporal
ALTER LOGIN login_sql_alumno WITH CHECK_EXPIRATION = ON;  -- Paso para forzar el estado
GO

-- 2C. Verificar que la expiración forzada se haya aplicado.
SELECT name, is_expiration_checked,
       is_disabled
FROM sys.sql_logins
WHERE name = 'login_sql_alumno';
GO
```

No se encontraron problemas.

Resultados | Mensajes

	name	is_policy_checked	is_expiration_checked
1	login_sql_alumno	1	1

	name	is_expiration_checked	is_disabled
1	login_sql_alumno	1	0

xp_cmdshell es un procedimiento almacenado extendido que permite a los usuarios (con los permisos adecuados) ejecutar comandos del sistema operativo (Windows o Linux) directamente desde T-SQL. Por seguridad, debe estar deshabilitado, a menos que sea estrictamente necesario.

```sql
-- Habilitar opciones avanzadas para cambiar la configuración
EXEC sp_configure 'show advanced options', 1;
RECONFIGURE;
GO

-- Deshabilitar el procedimiento almacenado extendido xp_cmdshell
EXEC sp_configure 'xp_cmdshell', 0;
RECONFIGURE;
GO

-- (Opcional) Deshabilitar opciones avanzadas de nuevo
EXEC sp_configure 'show advanced options', 0;
RECONFIGURE;
GO

-- Para documentar la configuración (demostrar que está deshabilitado):
SELECT name, value, value_in_use
FROM sys.configurations
WHERE name = 'xp_cmdshell';
```

76 %

No se encontraron problemas.

Resultados | Mensajes

	name	value	value_in_use
1	xp_cmdshell	0	0

El valor 1 indica que la característica para crear bases de datos contenidas está **Habilitada** a nivel de la instancia de SQL Server.

El valor NONE indica que la base de datos QhatuPeru **aún no permite** usuarios contenidos. Necesitas cambiar su configuración explícitamente.

```
-- 1. Revisar si la autenticación contenida está habilitada a nivel de instancia
SELECT name, value_in_use
FROM sys.configurations
WHERE name = 'contained database authentication';
GO

-- 2. Revisar si la base de datos QhatuPeru permite usuarios contenidos
USE master;
GO
SELECT name, containment_desc
FROM sys.databases
WHERE name = 'QhatuPeru';
GO
```

76 % No se encontraron problemas.

Resultados Mensajes

	name	value_in_use
1	contained database authentication	1

	name	containment_desc
1	QhatuPeru	NONE

Crear Credencial y Proxy para SQL Agent

Esta configuración permite a los Jobs (tareas programadas) de SQL Agent ejecutar comandos del OS (CmdExec) utilizando credenciales de baja prioridad (VICTUSHP\elvis) en lugar de usar la cuenta de servicio de SQL Server.

```sql
-- **IMPORTANTE:** Reemplaza 'Tu_Password_Secreta_Aqui' con la contraseña de tu usuario 'elvis'

-- A. Crear la Credencial (almacena el login y password de VICTUSHP\elvis)
USE master;
CREATE CREDENTIAL [Credencial_OS_Agente]
    WITH IDENTITY = 'VICTUSHP\elvis',
    SECRET = 'Tu_Password_Secreta_Aqui';
GO

-- B. Crear el Proxy de SQL Agent, vinculándolo a la credencial
EXEC msdb.dbo.sp_add_proxy
    @proxy_name = N'Proxy_CmdExec_Seguro',
    @credential_name = N'Credencial_OS_Agente',
    @description = N'Proxy seguro para ejecutar comandos OS via SQL Agent Jobs.';
GO

-- C. Habilitar el proxy para el subsistema CmdExec (ID 3)
EXEC msdb.dbo.sp_grant_proxy_to_subsystem
    @proxy_name = N'Proxy_CmdExec_Seguro',
    @subsystem_id = 3;
GO
```

3

Preparación y Creación del Rol Personalizado

Si el rol ventas_readwrite o las tablas de prueba no existían, estos comandos las crean.

```sql
USE QhatuPeru;
GO

-- 1A. Creación de las tablas de prueba (si no existen)
IF OBJECT_ID('GUIA_ENVIO') IS NULL
BEGIN
    CREATE TABLE GUIA_ENVIO (GuiaID INT PRIMARY KEY IDENTITY(1,1), Direccion NVARCHAR(200));
END
IF OBJECT_ID('GUIA_DETALLE') IS NULL
BEGIN
    CREATE TABLE GUIA_DETALLE (DetalleID INT PRIMARY KEY IDENTITY(1,1), GuiaID INT);
END
GO

-- 1B. Crear el rol de base de datos personalizado (si no existe)
IF DATABASE_PRINCIPAL_ID('ventas_readwrite') IS NULL
BEGIN
    CREATE ROLE ventas_readwrite;
END
GO
```

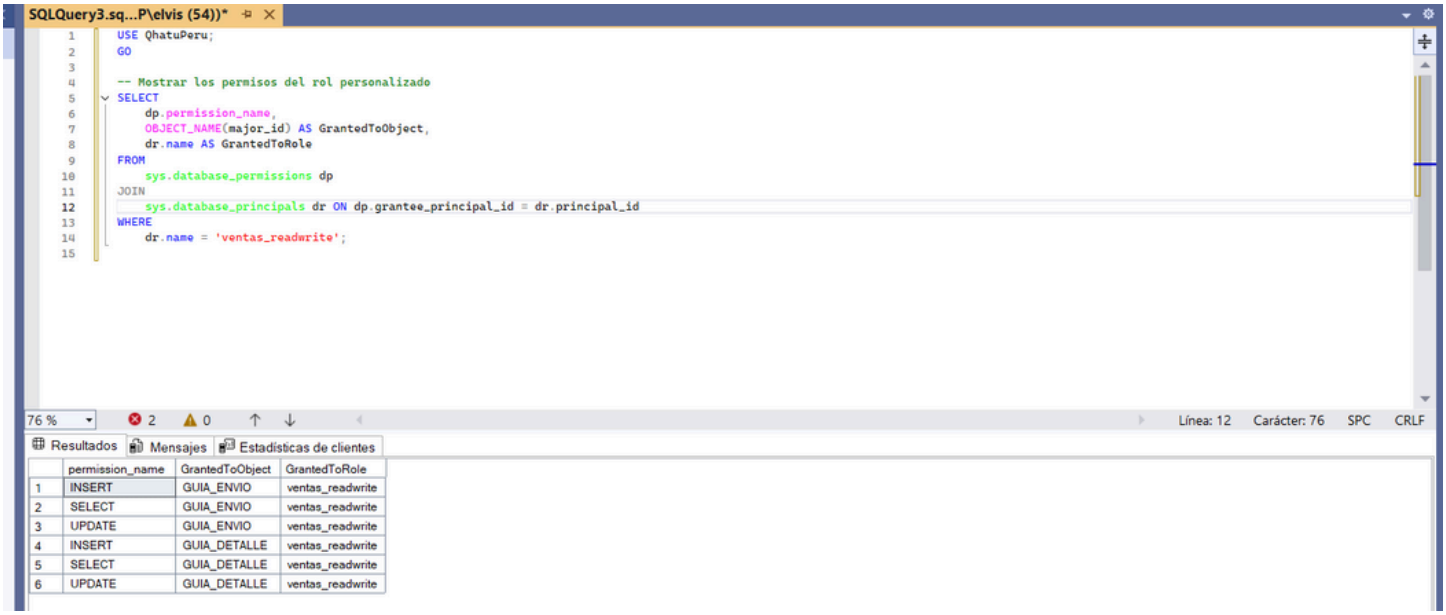
No se encontraron problemas.

Mensajes

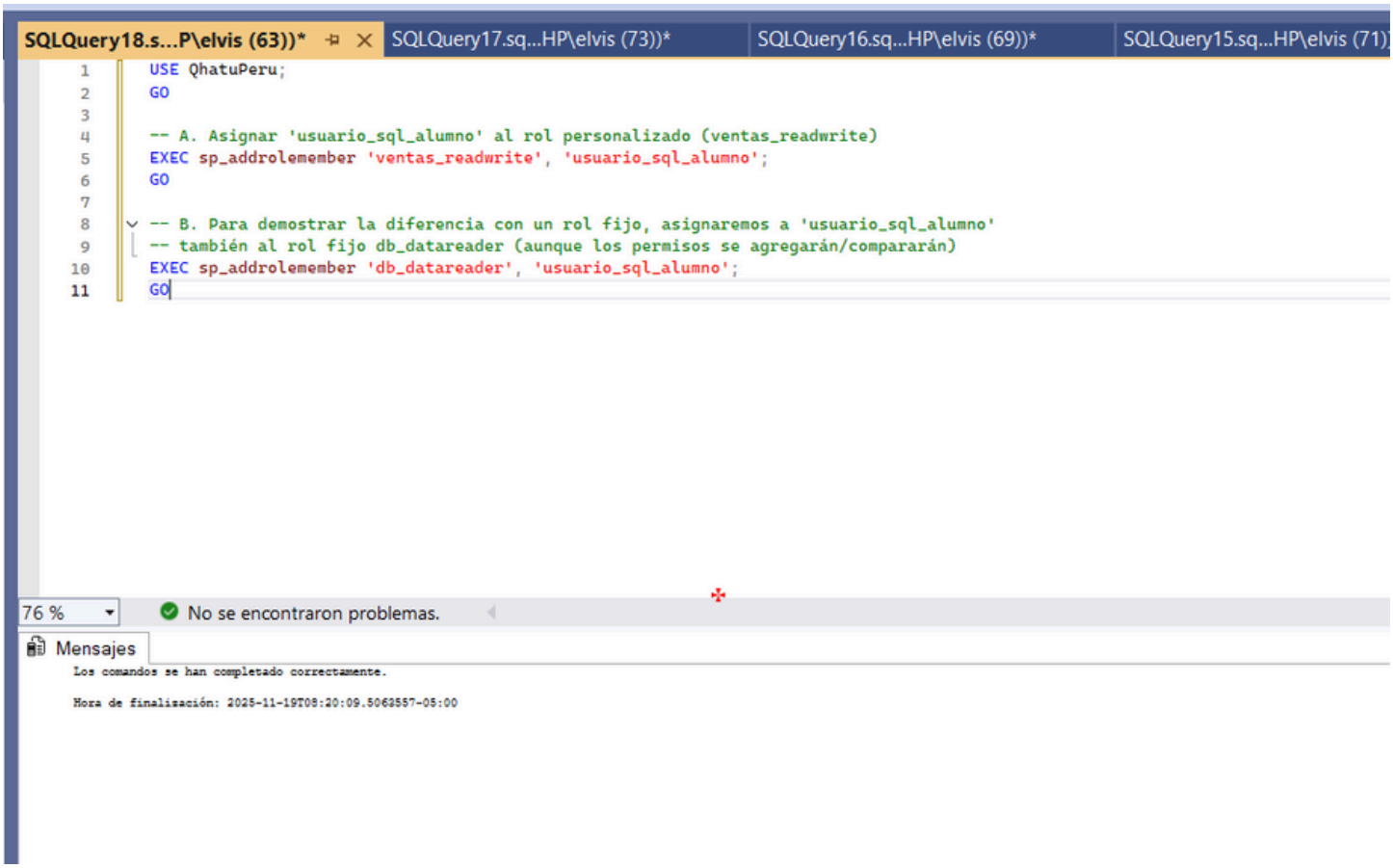
```
Los comandos se han completado correctamente.

Hora de finalización: 2025-11-19T08:18:32.0546560-05:00
```

Otorgar Permisos Granulares

Este paso otorga los permisos específicos (SELECT/INSERT/UPDATE) requeridos por el rol personalizado **ventas_readwrite**.

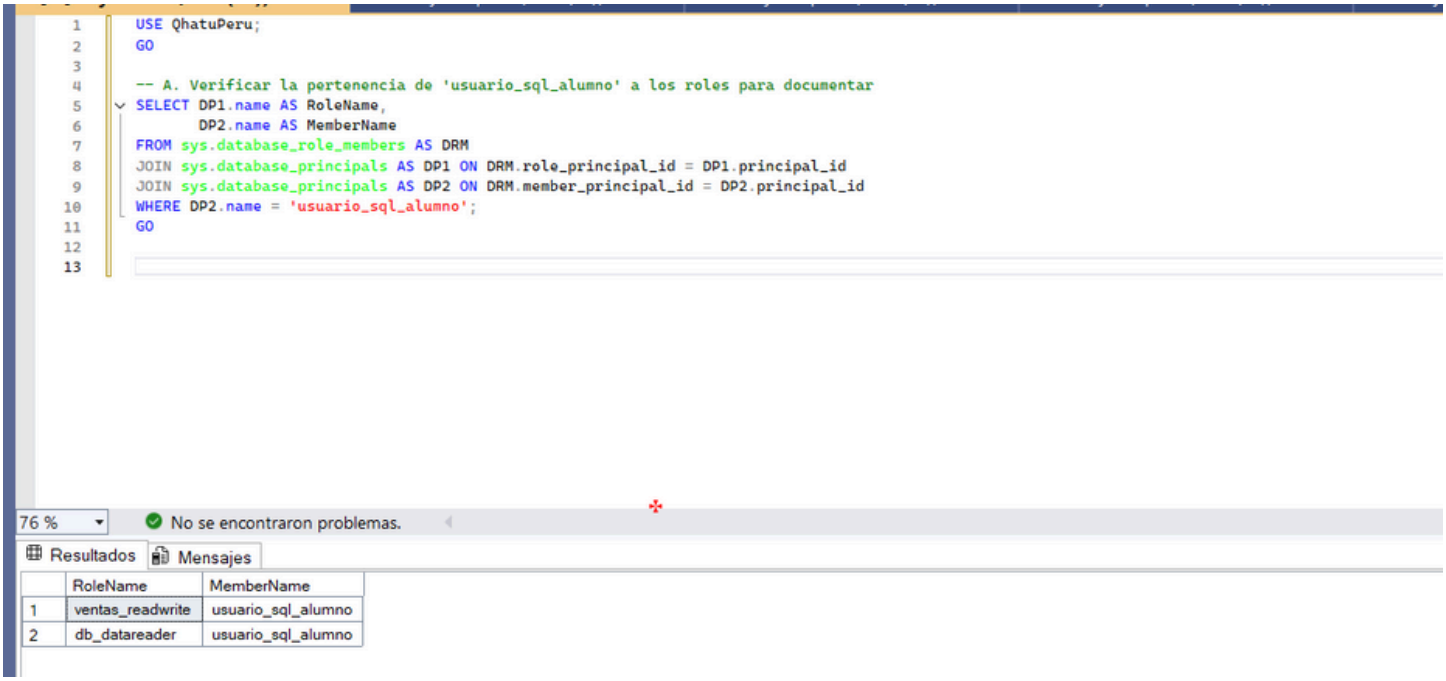


Asignamos el usuario de SQL (usuario_sql_alumno) al rol personalizado. No podemos usar tu usuario de Windows (VICTUSHP\elvis) para la comparación porque es el dbo (administrador) y tiene permisos totales.



Mostrar Diferencias y Documentación

Esta sección documenta la pertenencia a los roles y explica la diferencia clave requerida por el ejercicio.



```sql
USE QhatuPeru;
GO

-- A. Verificar la pertenencia de 'usuario_sql_alumno' a los roles para documentar
SELECT DP1.name AS RoleName,
       DP2.name AS MemberName
FROM sys.database_role_members AS DRM
JOIN sys.database_principals AS DP1 ON DRM.role_principal_id = DP1.principal_id
JOIN sys.database_principals AS DP2 ON DRM.member_principal_id = DP2.principal_id
WHERE DP2.name = 'usuario_sql_alumno';
GO
```

76 % — No se encontraron problemas.

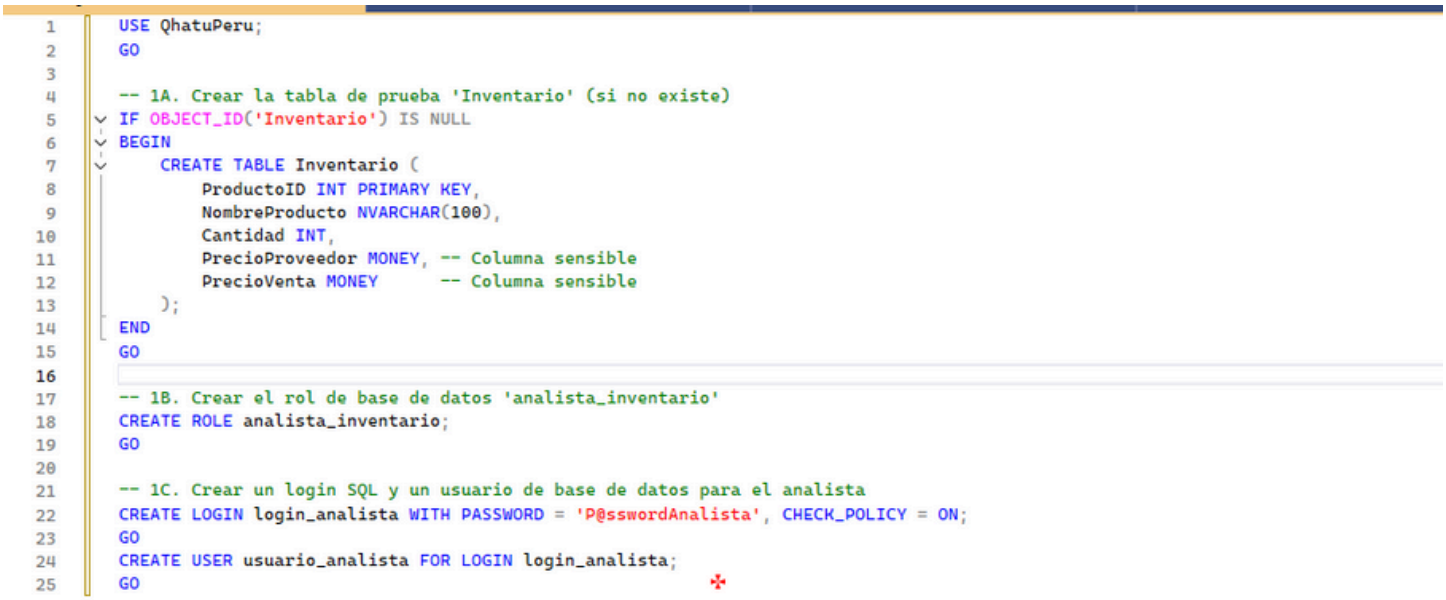
Resultados | Mensajes

	RoleName	MemberName
1	ventas_readwrite	usuario_sql_alumno
2	db_datareader	usuario_sql_alumno

4

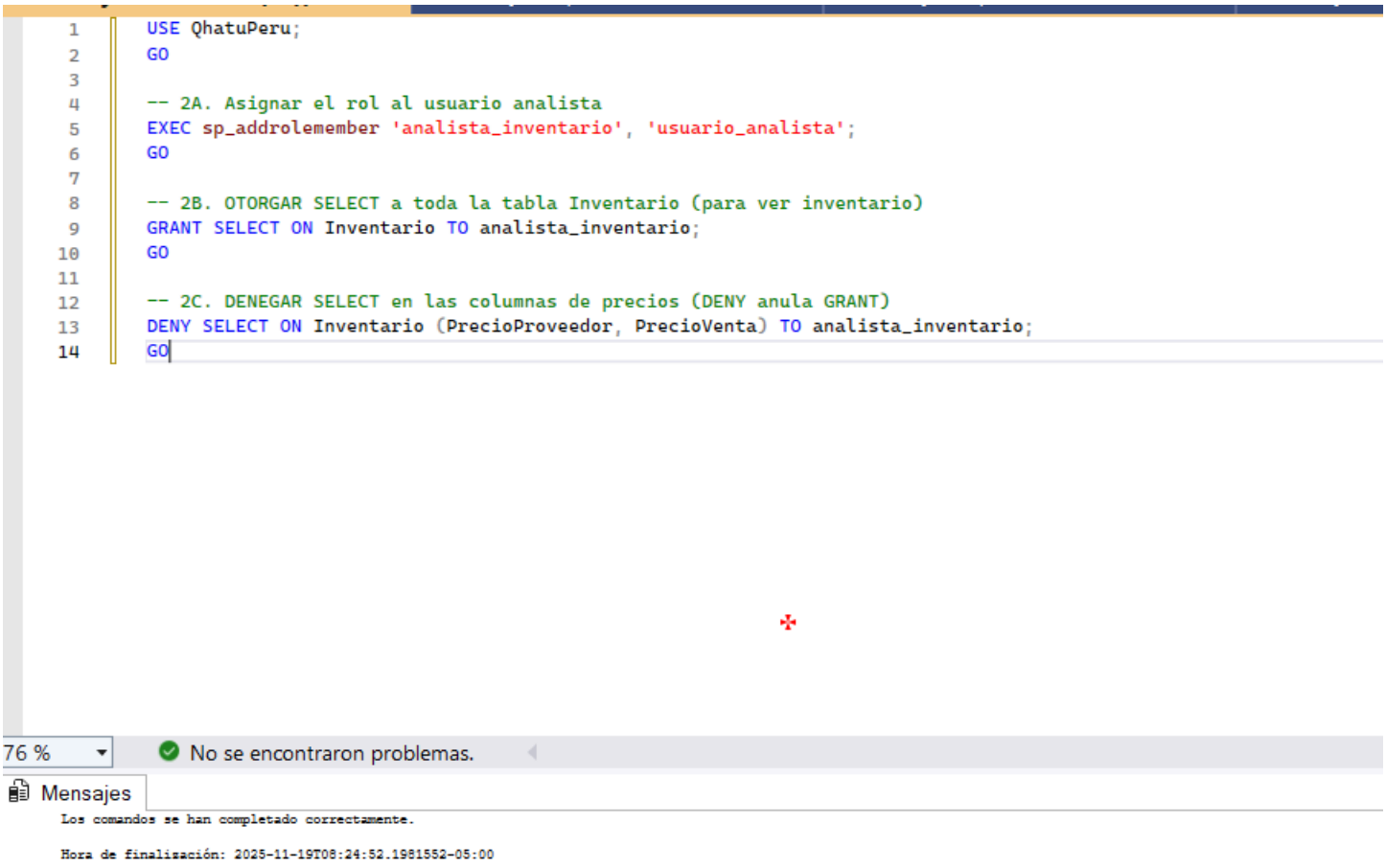
Preparación: Creación de Objetos de Prueba

Necesitamos una tabla, un *login* y un *rol* para la demostración.



```sql
USE QhatuPeru;
GO

-- 1A. Crear la tabla de prueba 'Inventario' (si no existe)
IF OBJECT_ID('Inventario') IS NULL
BEGIN
    CREATE TABLE Inventario (
        ProductoID INT PRIMARY KEY,
        NombreProducto NVARCHAR(100),
        Cantidad INT,
        PrecioProveedor MONEY, -- Columna sensible
        PrecioVenta MONEY      -- Columna sensible
    );
END
GO

-- 1B. Crear el rol de base de datos 'analista_inventario'
CREATE ROLE analista_inventario;
GO

-- 1C. Crear un login SQL y un usuario de base de datos para el analista
CREATE LOGIN login_analista WITH PASSWORD = 'P@sswordAnalista', CHECK_POLICY = ON;
GO
CREATE USER usuario_analista FOR LOGIN login_analista;
GO
```

Aquí aplicamos la lógica: se **otorga** la lectura a la tabla completa (GRANT SELECT) y luego se **deniega** explícitamente la lectura de las columnas sensibles (DENY SELECT).



Demostración del Control de Acceso (Prueba)

Para documentar el proyecto, ejecute esta sección para mostrar la diferencia entre una consulta permitida y una negada.

SQL

```sql
USE QhatuPeru;
GO

-- 3A. Simular ser el 'usuario_analista'
EXECUTE AS USER = 'usuario_analista';
GO

-- PRUEBA 1 (DEBE FUNCIONAR): Ver columnas no sensibles (Cantidad, Nombre)
-- Demuestra que el analista puede ver el inventario.
SELECT ProductoID, NombreProducto, Cantidad
FROM Inventario;
GO

-- PRUEBA 2 (DEBE FALLAR): Intentar ver columnas sensibles (Precios)
-- Esto DEBE generar un error de permisos (porque DENY tiene precedencia).
SELECT ProductoID, NombreProducto, PrecioProveedor, PrecioVenta
FROM Inventario;
GO

-- 3B. Volver al usuario administrador
REVERT;
GO
```

76 % 9 0

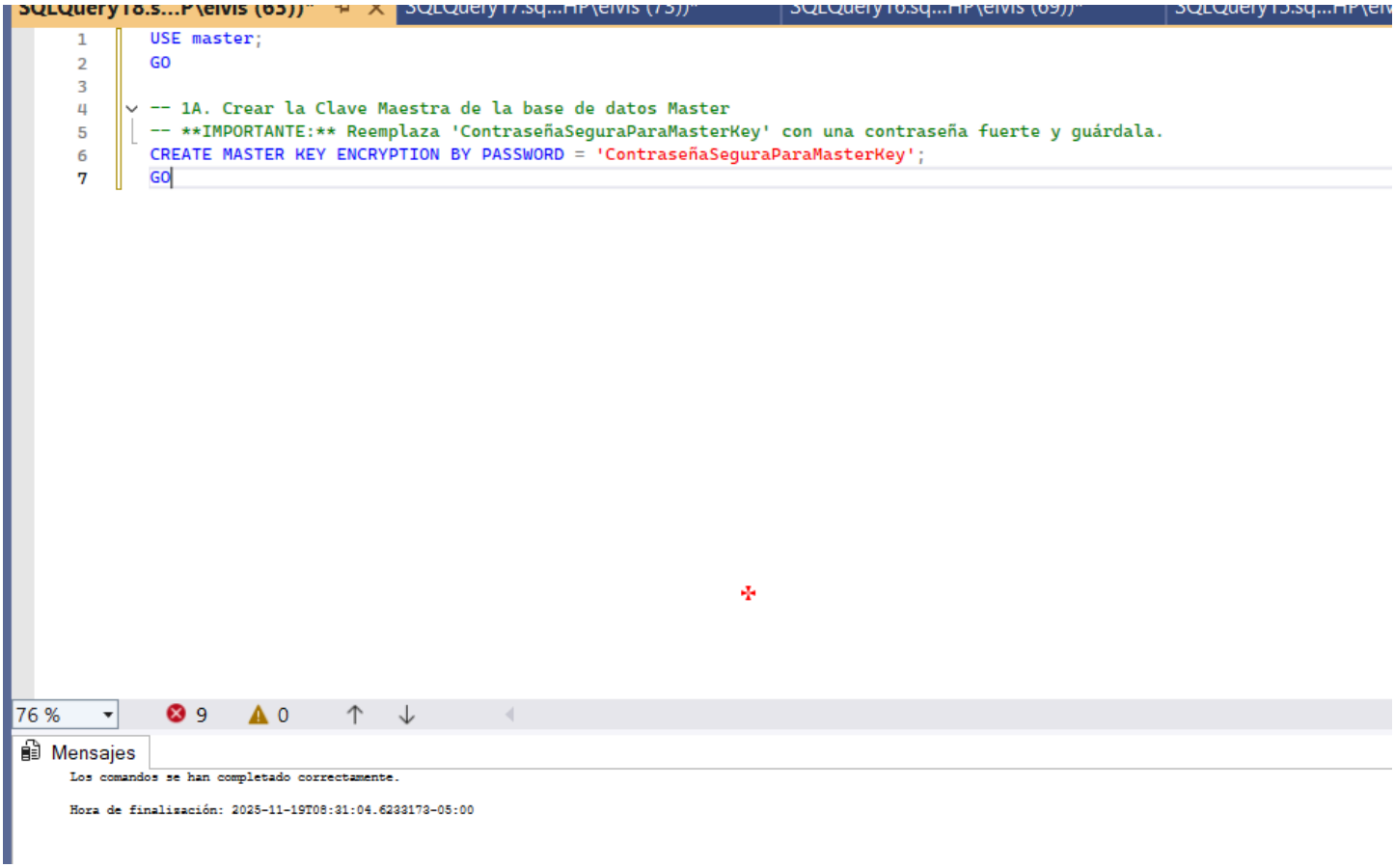
Resultados Mensajes

```
(0 filas afectadas)
Mens. 230, Nivel 14, Estado 1, Línea 16
The SELECT permission was denied on the column 'PrecioProveedor' of the object 'Inventario', database 'QhatuPeru', schema 'dbo'.
Mens. 230, Nivel 14, Estado 1, Línea 16
The SELECT permission was denied on the column 'PrecioVenta' of the object 'Inventario', database 'QhatuPeru', schema 'dbo'.

Hora de finalización: 2025-11-19T08:25:56.1415405-05:00
```

5

Crear la Master Key del Servidor

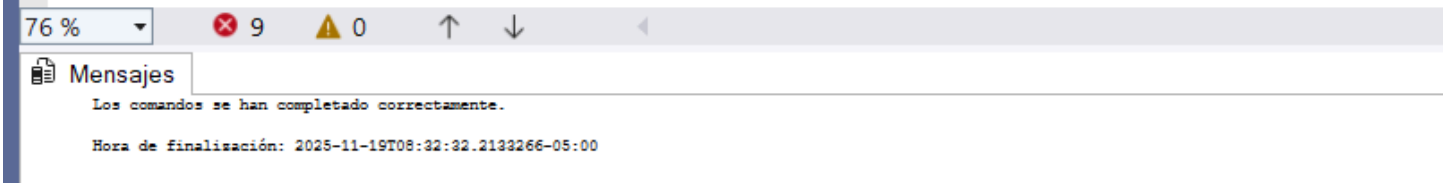
Este paso crea la clave maestra que protege el certificado. Se debe ejecutar en la base de datos master.



Crear el Certificado de Servidor

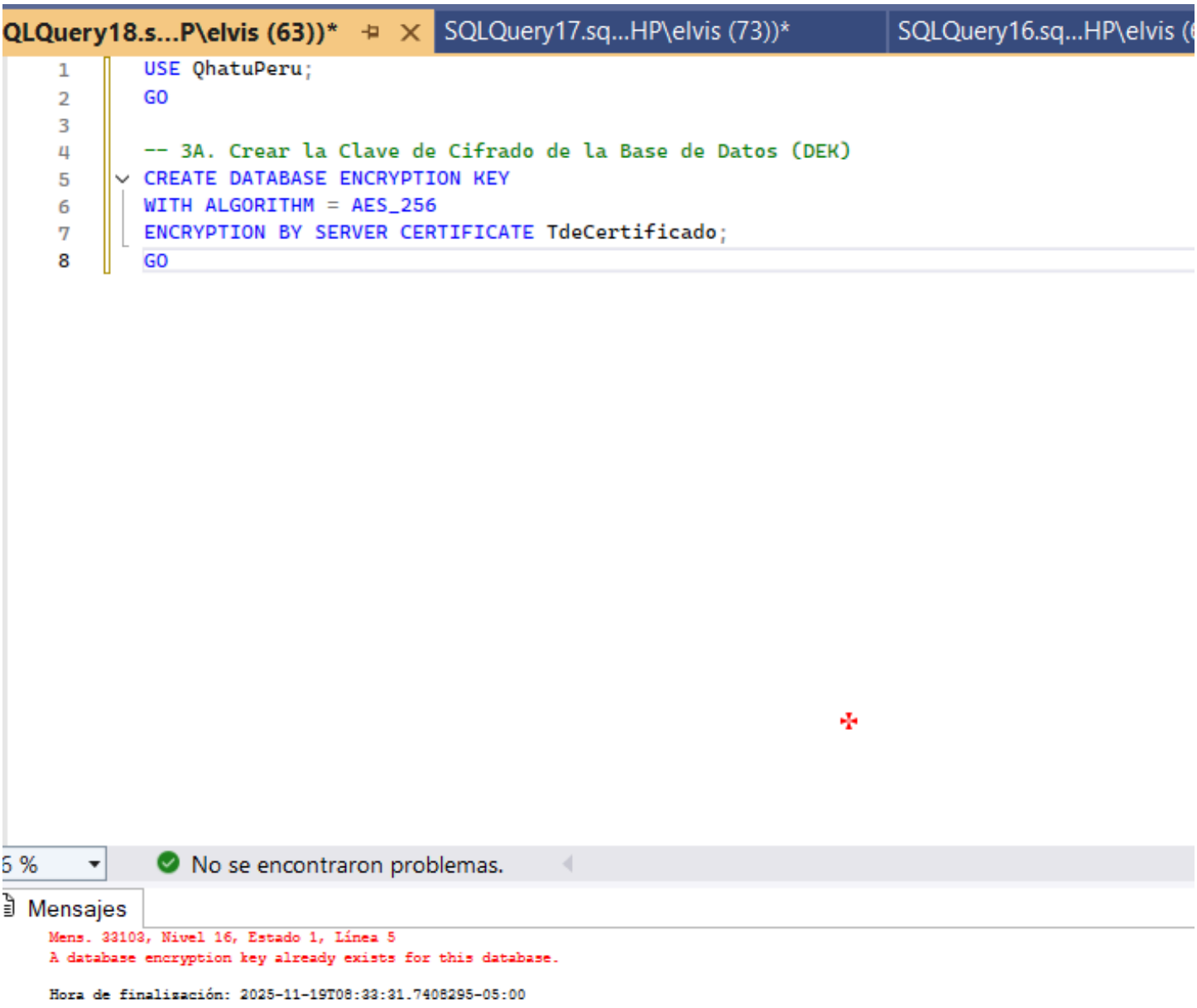
Este certificado, creado a partir de la Master Key, protegerá la clave de cifrado de la base de datos. Se ejecuta en master.

```sql
USE master;
GO

-- 2A. Crear el Certificado del Servidor
CREATE CERTIFICATE TdeCertificado
   WITH SUBJECT = 'Certificado para TDE en QhatuPeru';
GO
```

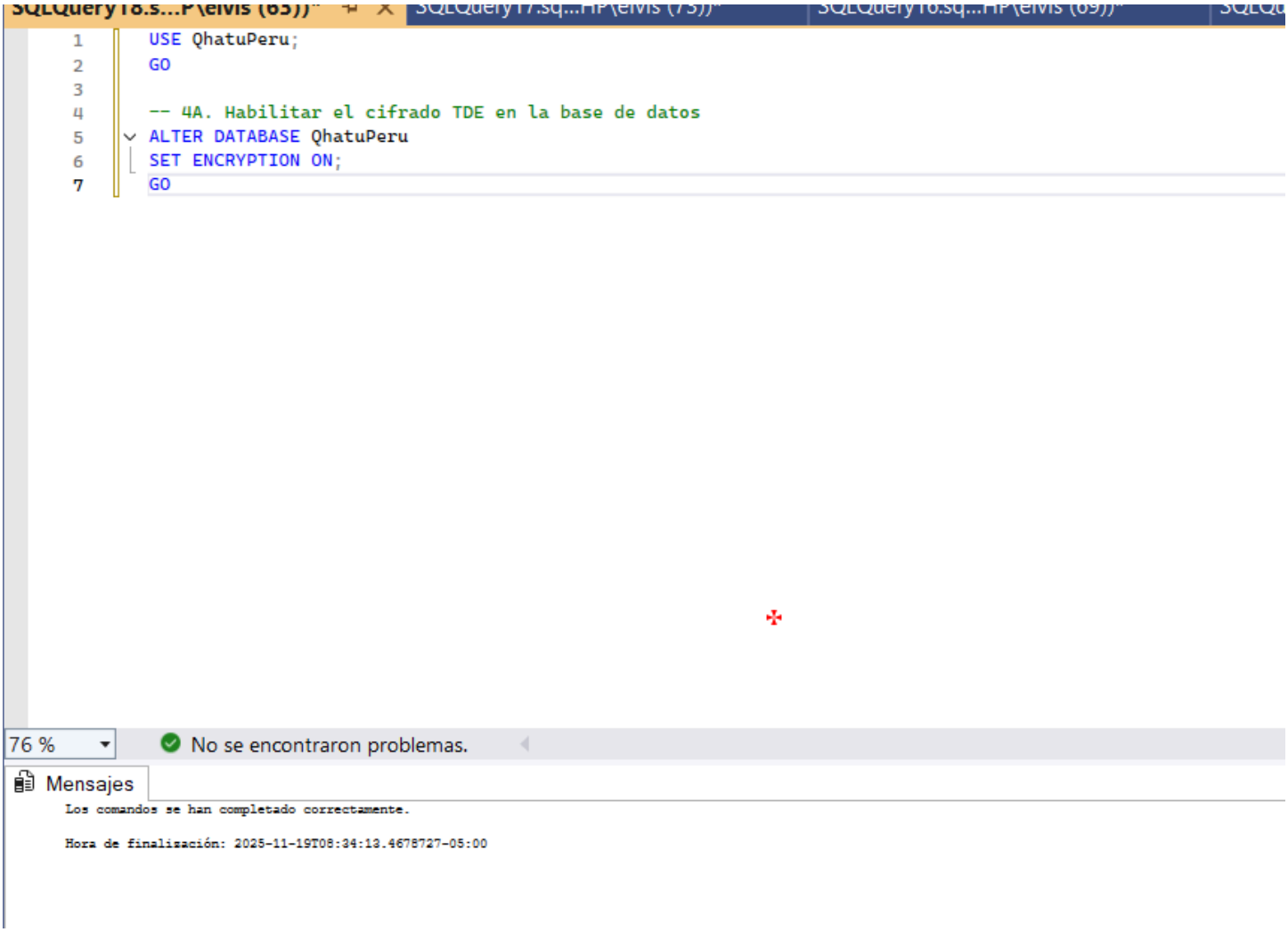


Crear la Clave de Cifrado de la Base de Datos (DEK)

Este es el primer paso específico en la base de datos QhatuPeru. La DEK es la clave simétrica real que cifrará los datos.



Finalmente, se activa el cifrado en la base de datos QhatuPeru. El proceso de cifrado de los archivos físicos (MDF/LDF) comenzará automáticamente.



Para demostrar que TDE se ha implementado correctamente, utiliza esta consulta para verificar el estado de cifrado:

```sql
-- 5A. Verificar el estado del cifrado (Encryption State)
-- Estado 2 = Encrypted (Completado)
-- Estado 3 = Encryption in progress (En progreso)
SELECT DB_NAME(database_id) AS DatabaseName,
       encryption_state_desc
FROM sys.dm_database_encryption_keys
WHERE DB_NAME(database_id) = 'QhatuPeru';
GO
```

76 % 1 0

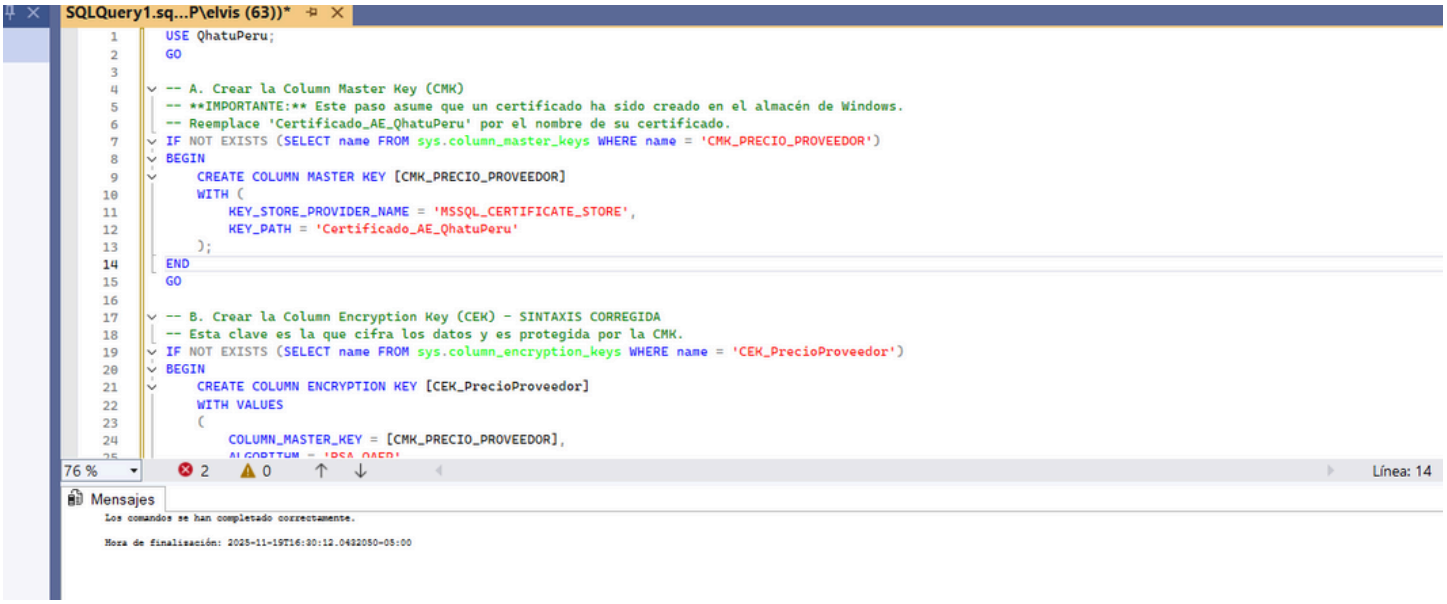
Resultados Mensajes

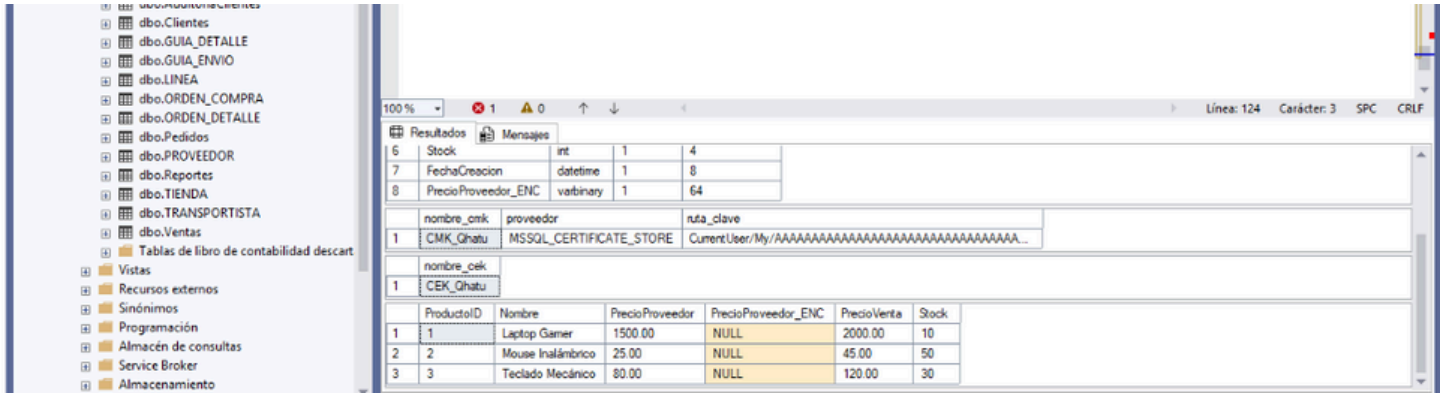
	DatabaseName	encryption_state_desc
1	QhatuPeru	ENCRYPTED

6

Creación de las Claves de Cifrado (CMK y CEK)

La Columna Master Key (CMK) protege la Columna Encryption Key (CEK), y ambas se necesitan antes de cifrar una columna.





7 El objetivo es **configurar una auditoría de seguridad** que consta de dos partes:

1. **Un Server Audit (Auditoría de Servidor):** Debe registrar tanto los intentos de inicio de sesión (**login**) que son **fallidos** como los que son **exitosos**.
2. **Un Database Audit Specification (Especificación de Auditoría de Base de Datos):** Esta debe configurarse específicamente para la base de datos **QhatuPeru** y tiene que registrar los cambios en la estructura de la base de datos (lenguaje DDL) como son las operaciones de **CREATE**, **ALTER**, y **DROP** (crear, modificar o eliminar objetos) aplicados a los objetos considerados **críticos**.

```
BEGIN
    CREATE DATABASE AUDIT SPECIFICATION EspecificacionBDQhatu
    FOR SERVER AUDIT AuditoriaQhatu
    ADD (SELECT, INSERT, UPDATE, DELETE ON DATABASE::QhatuPeru BY PUBLIC),
    ADD (EXECUTE ON DATABASE::QhatuPeru BY PUBLIC)
    WITH (STATE = ON);
    PRINT 'Especificación de BD creada';
END
ELSE
    PRINT 'Especificación de BD ya existe';

PRINT '☑ Auditoría configurada exitosamente';
GO
```

```
Auditoría creada en Application Log
Auditoría habilitada
Especificación de servidor creada
Especificación de BD creada
? Auditoría configurada exitosamente

Hora de finalización: 2025-11-19T12:28:51.8573098-05:00
```

8 **Objetivo:** Configurar una sesión de **Extended Events (Eventos Extendidos)**.

- **Funcionalidad:** Esta sesión debe **capturar** eventos de **deadlocks** (bloqueos mutuos) y eventos de **login failed** (intentos de inicio de sesión fallidos).
- **Resultado esperado:** Los eventos deben **guardarse en un archivo**, y se debe crear una **vista** que permita consultar y analizar esos datos de **XEvent** directamente desde la base de datos.

```
USE QhatuPeru;
GO

-- 1. Crear una función para simplificar la lectura de archivos .xel
CREATE FUNCTION dbo.fn_ReadXEvents (@FilePath NVARCHAR(4000))
RETURNS TABLE
AS
RETURN
(
    -- sys.fn_xe_file_target_read_file lee los archivos de eventos y devuelve el contenido XML
    SELECT
        CAST(event_data AS XML) AS EventXML
    FROM sys.fn_xe_file_target_read_file(@FilePath, NULL, NULL, NULL)
);
GO

-- 2. Crear la vista que consulta los eventos y extrae datos clave
CREATE VIEW Monitoreo_XE_QhatuPeru AS
SELECT
    -- Extraer el nombre del evento
    T.EventXML.value('(/event/@name)[1]', 'NVARCHAR(100)') AS EventName,
    -- Extraer la marca de tiempo del evento
    T.EventXML.value('(/event/@timestamp)[1]', 'datetimeoffset') AS EventTime,
    -- Intentar extraer el nombre de la BD (disponible en deadlocks)
```

```
Vista Monitoreo_XE_QhatuPeru creada. Consulta SELECT * FROM Monitoreo_XE_QhatuPeru;

Hora de finalización: 2025-11-19T17:37:48.2738928-05:00
```

9 **Objetivo:** Implementar técnicas de ocultación de datos y control de acceso.

Funcionalidad 1 (Dynamic Data Masking): Aplicar **Dynamic Data Masking (Enmascaramiento Dinámico de Datos)** a columnas que contengan información sensible (el ejemplo dado es el número de **Teléfono en la tabla PROVEEDOR**), para que el dato real no sea visible para la mayoría de los usuarios.

Funcionalidad 2 (Acceso Controlado): Crear una **vista segura** que permita a un conjunto limitado de usuarios (validado mediante una **función que valide su rol**) ver los **datos completos y sin enmascarar**.

SQLQuery7.sq...P\elvis (79))* | SQLQuery6.sql ...HP\elvis (78))* | SQLQuery5.sql ...HP

```
1  USE QhatuPeru;
2  GO
3
4  -- 1. Crear un rol para usuarios privilegiados
5  CREATE ROLE Rol_Acceso_Telefono_Completo;
6  GO
7
8  -- 2. Asignar el permiso UNMASK al nuevo rol
9  -- Este es el permiso que anula el DDM
10 GRANT UNMASK TO Rol_Acceso_Telefono_Completo;
11 GO
12
13 -- 3. Crear la Vista Segura (que selecciona directamente la columna)
14 -- Esta vista simplemente selecciona el campo, el enmascaramiento se aplica o no
15 -- automáticamente basado en el permiso UNMASK del usuario.
16 CREATE VIEW Proveedor_Detalle_Telefono AS
17 SELECT
18     ProveedorID,
19     Nombre,
20     Telefono -- Esta columna se mostrará sin máscara si el usuario tiene permiso
21 FROM
22     PROVEEDOR;
23 GO
```

76 % | 2 | 0

Mensajes

```
Los comandos se han completado correctamente.

Hora de finalización: 2025-11-19T17:39:22.3518585-05:00
```

10 **Objetivo:** Este es un proyecto integrador que combina múltiples características de seguridad avanzadas.

Pasos:

1. Crear un **rol auditor_seguridad** (para segregación de funciones).
2. **Habilitar TDE (Transparent Data Encryption)** en la base de datos (si no está habilitado).

3. **Preparar Always Encrypted** para una **columna sensible** (cifrado persistente de la columna).
4. **Configurar auditoría** para una tabla específica (auditar el acceso a esa tabla).
5. Dejar un **procedimiento almacenado** que registre cambios críticos (esto sugiere una **traza** o registro de auditoría personalizado).

	Categoria	Elemento	Estado
1	ESTADO SEGURIDAD QHATU PERU	TDE	HABILITADO
2	ESTADO SEGURIDAD QHATU PERU	ALWAYS ENCRYPTED	HABILITADO
3	ESTADO SEGURIDAD QHATU PERU	AUDITORÍA SERVER	ACTIVA
4	ESTADO SEGURIDAD QHATU PERU	ENMASCARAMIENTO DINÁMICO	ACTIVO