**TRIGGERS (disparadores)   
¿Qué son?**

Tipo de Proceso Almacenado que se activa cuando se producen eventos en las TABLAS.

* Siempre están asociados a una tabla.
* No reciben ni retornan parámetros.
* A diferencia de los Procesos Almacenados se ejecutan automáticamente cuando se realiza la instrucción a la que está asociado y no pueden ser llamados.
* Suelen ser utilizados para tareas de mantenimiento o administración de las BBDD.

**SQL Server**

Hay dos tipos de TRIGGERS:

* DML: Data Manipulation Language – asociados a INSERT, UPDATE O DELETE – NOSOTROS VAMOS A VER ESTOS
* DDL: Data Defenition Language – asociados a CREATE, ALTER y DROP

**TRIGGERS del tipo DML**

En estos TRIGGERS tenemos dos tablas temporales que se crean cuando son activados: INSERTED y DELETED, y su estructura es igual a la de la tabla a la que está asociada el TRIGGER.

* INSERTED – funciona con INSERT y UPDATE, se utilizan para acceder a la información después.
* DELETED – funciona con UPDATE Y DELETE, se utilizan para acceder a la información antes de ser modificada o eliminada.

**SINTAXIS**

CREATE TRIGGER triggerName\*\* –creamos el trigger

ON tableName –indicamos a que tabla lo asociamos

FOR|AFTER|INSTEAD OF INSERT,DELETE,UPDATE –función del trigger y evento sobre el que se ejecuta

AS –a partir de aquí ponemos las condiciones o acciones que realizara el trigger

SET NOCOUNT ON –sirve para que no se muestre en la consola la acción del trigger(es opcional pero recomendado)

CONDITIONS or STATEMENTS

GO

-- INSTEAD OF: pueden crearse uno por acción y no están permitidos en tablas que son objetivo de FK con tipo cascada.

-- FOR y AFTER: pueden crearse muchos por acción y no tienen restricciones.

**Acciones sobre TRIGGERS:**

Habilitarlos o deshabilitarlos:

ENABLE TRIGGER trAlumnosUpdate ON Talumnos;

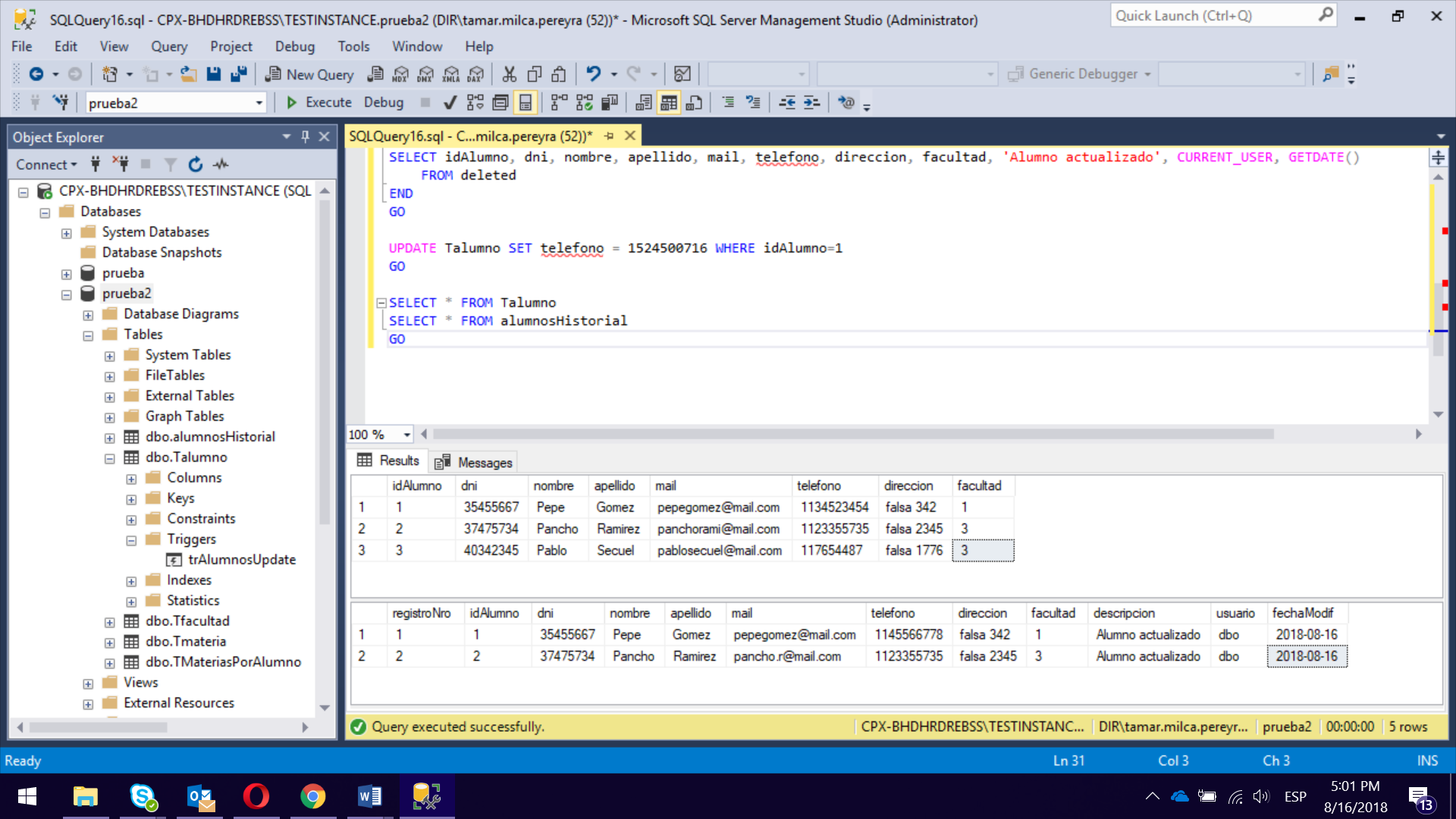
DISABLE TRIGGER trAlumnosUpdate ON Talumnos;

Eliminarlos:

DROP TRIGGER IF EXISTS trAlumnosUpdate;

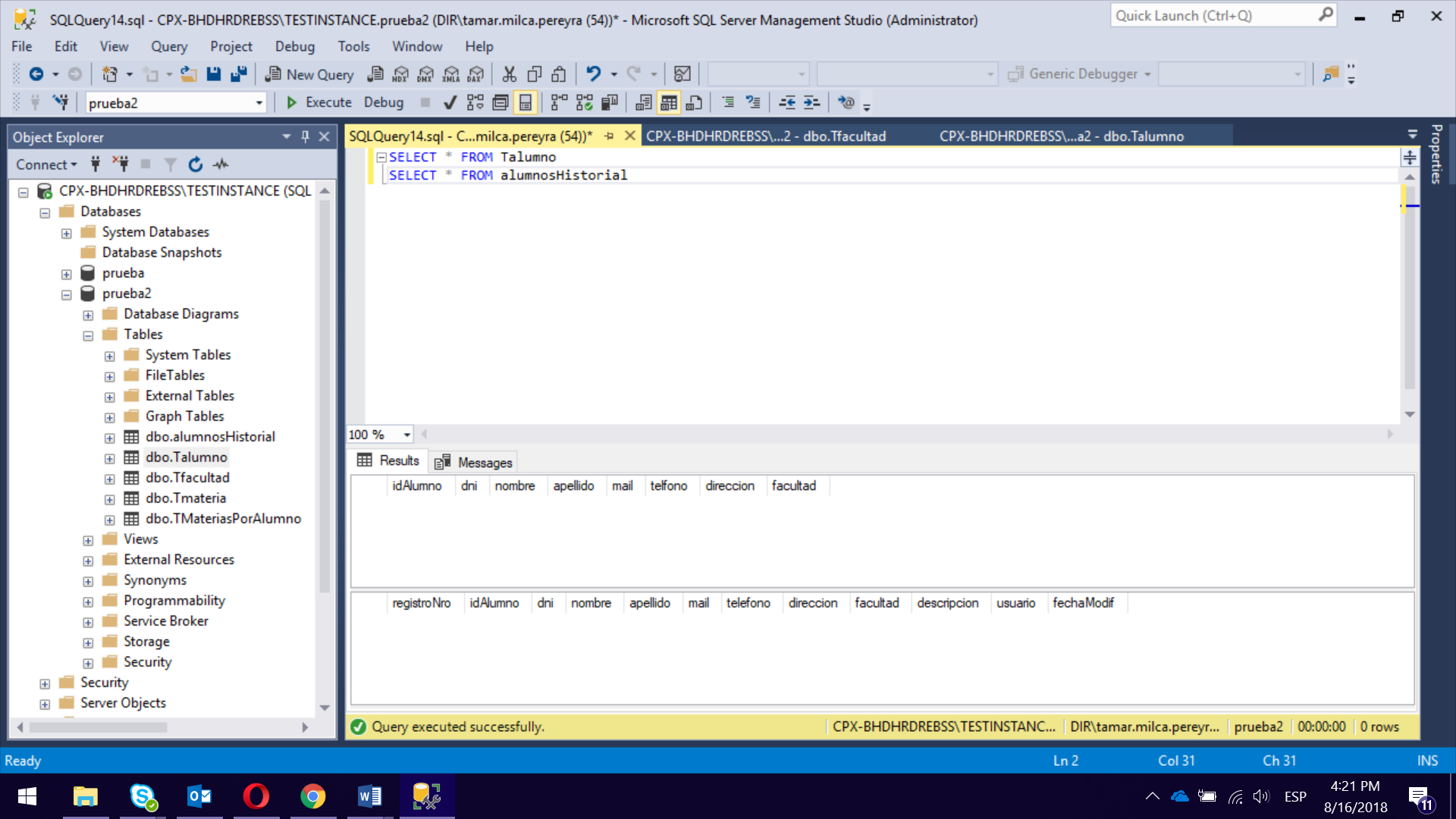
**Ejemplo**

Suponiendo que tenemos una tabla de alumnos **Talumno**



Creamos la tabla donde registraremos todo tipo de modificaciones, INSERT, UPDATE o DELETE

CREATE TABLE alumnosHistorial (registroNro INT PRIMARY KEY IDENTITY(1,1), idAlumno INT, dni INT, nombre VARCHAR (50), apellido VARCHAR (50), mail VARCHAR (50), telefono VARCHAR (20), direccion VARCHAR(50), facultad INT, descripcion VARCHAR (20), usuario VARCHAR (15), fechaModif DATE)



**TRIGGER de ACTUALIZACION**

--Procederemos a crear el TRIGGER

CREATE TRIGGER trAlumnosUpdate

ON Talumno FOR UPDATE

AS BEGIN

SET NOCOUNT ON;

INSERT INTO alumnosHistorial (dni, nombre, apellido, mail, telefono, direccion, facultad, descripcion, usuario, fechaModif)

SELECT dni, nombre, apellido, mail, telefono, direccion, facultad, 'Alumno actualizado', CURRENT\_USER, GETDATE()

FROM **deleted**

END

GO

--Ejecutamos la accion

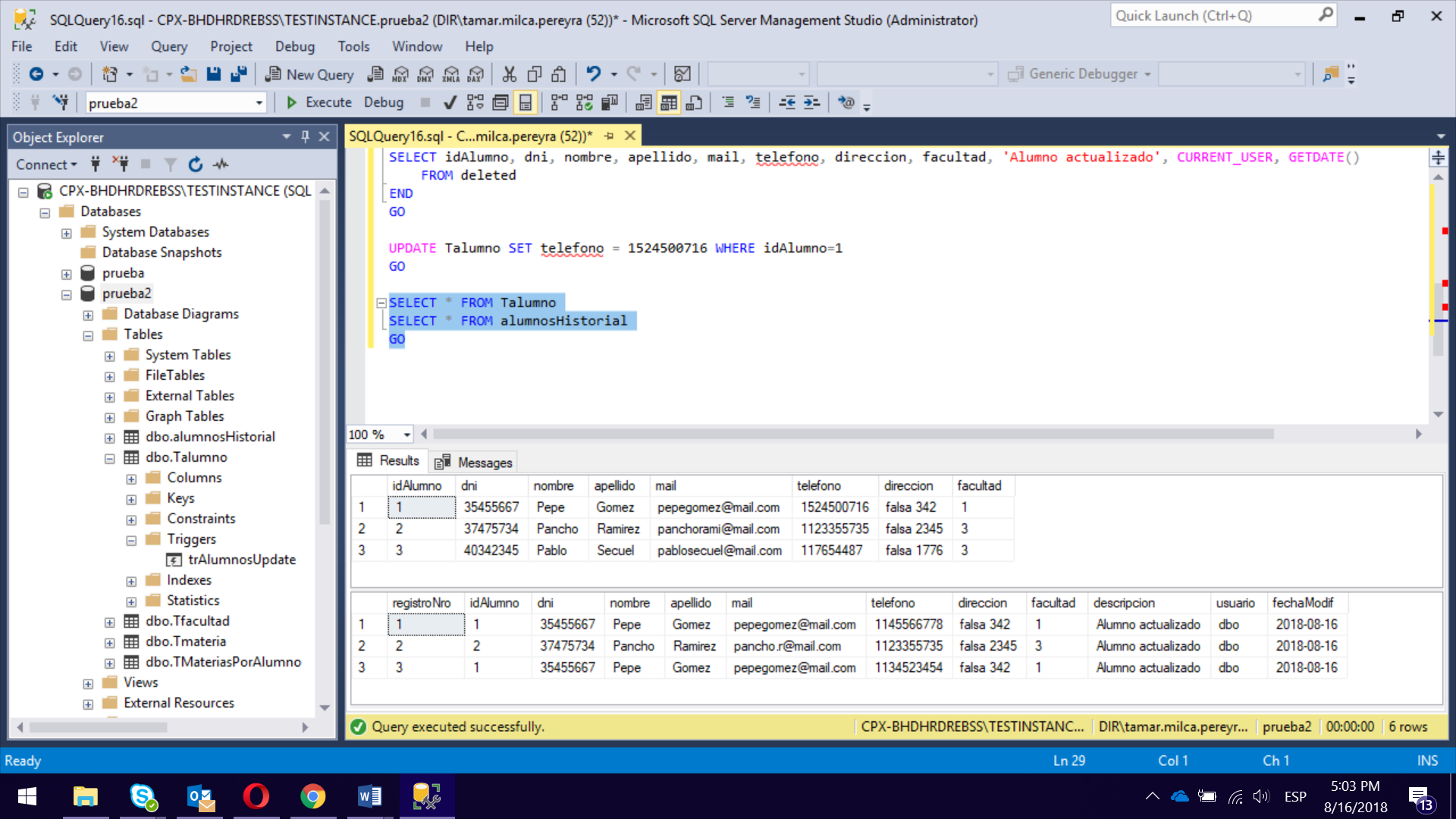
UPDATE Talumno SET telefono = **1524500716** WHERE idAlumno=1;

--Corroboramos que funciono bien

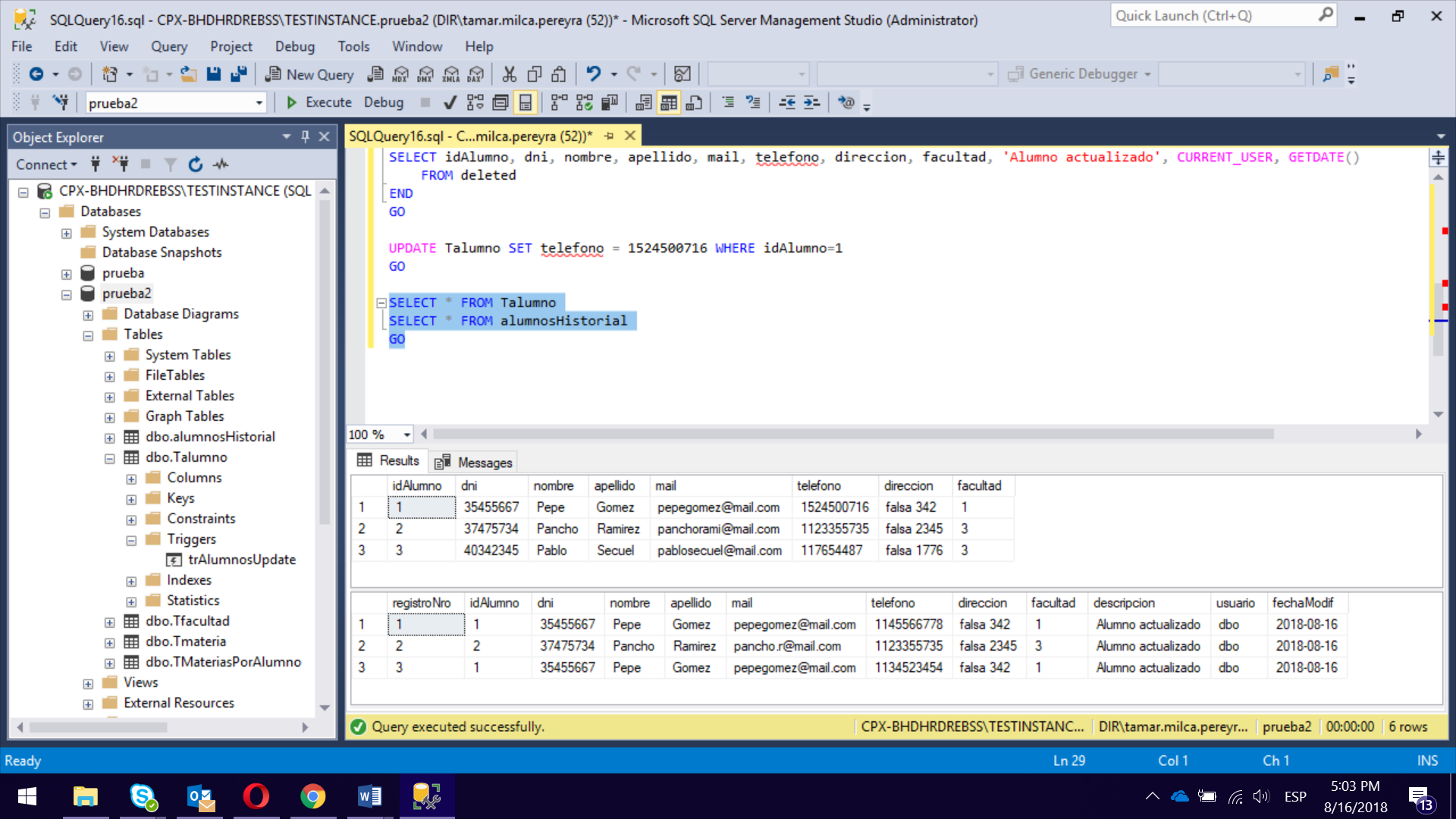
select \* from Talumno where idAlumno=1;

select \* from alumnosHistorial where idAlumno=1;

**Talumno**



**alumnosHistorial**



**TRIGGER de ELIMINACION**

--Procederemos a crear el TRIGGER

CREATE TRIGGER trAlumnosDeleteIF ON Talumno FOR DELETE

AS

SET NOCOUNT ON

IF (select idAlumno from deleted) IS NOT NULL

BEGIN

INSERT INTO alumnosHistorial (idAlumno, dni, nombre, apellido, mail, telefono,

direccion, facultad, descripcion, usuario, fechaModif)

SELECT idAlumno, dni, nombre, apellido, mail, telefono, direccion, facultad, 'Alumno eliminado’, CURRENT\_USER, GETDATE() FROM deleted

END

ELSE PRINT('NO EXISTE EL ALUMNO')

GO

--Una vez creado todo realizamos la QUERY de DELETE

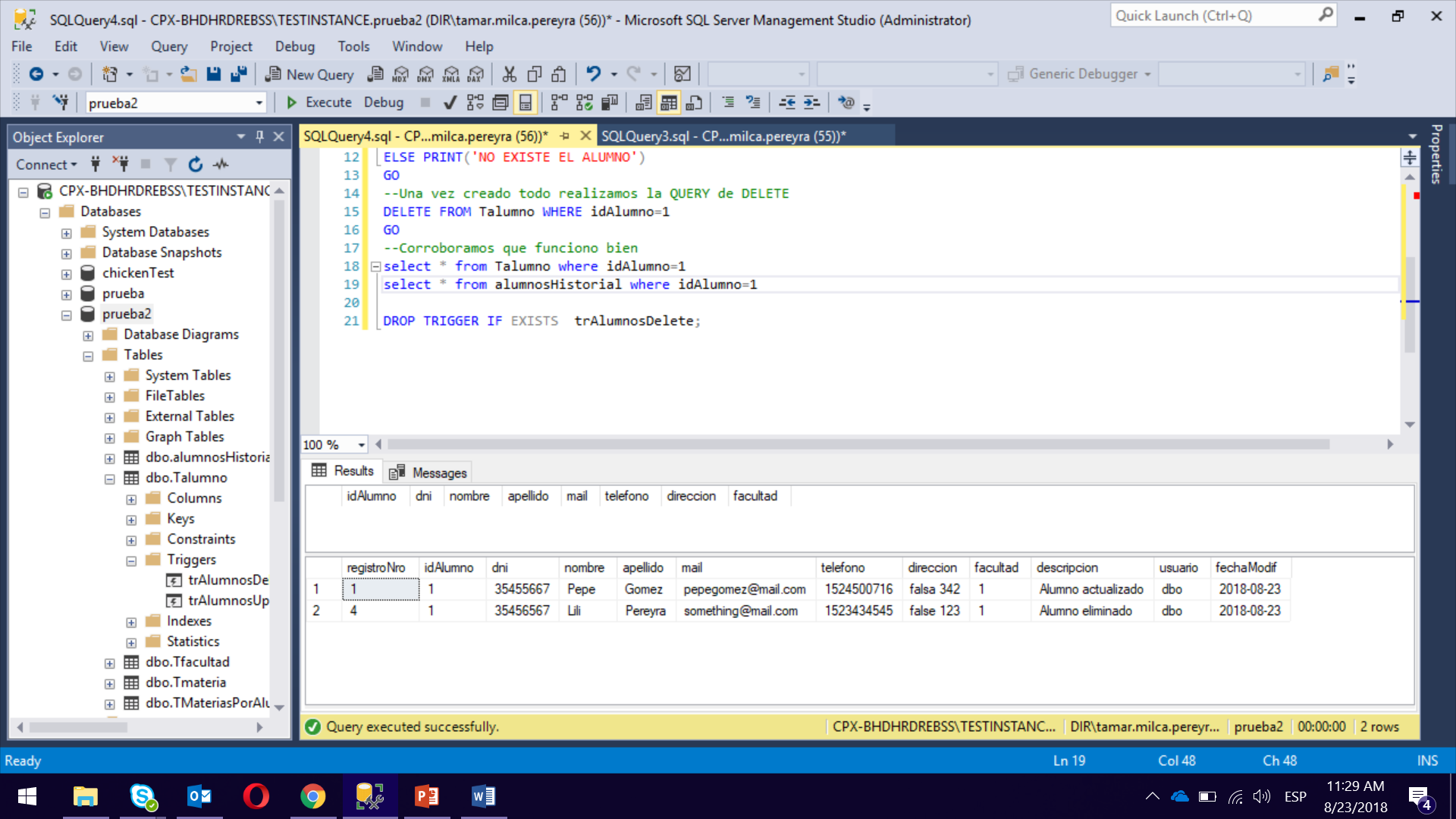
DELETE FROM Talumno WHERE idAlumno=1;

--Corroboramos que funciono bien

select \* from Talumno where idAlumno=1;

select \* from alumnosHistorial where idAlumno=1;

**Talumno**



**alumnosHistorial**

