**TECNOLOGICO NACIONAL DE MEXICO CAMPUS CULIACAN**

**INGENIERIA EN SISTEMAS COMPUTACIONALES**



**MATERIA**

ADMINISTRACION DE BASE DE DATOS

**INTEGRANTES**

LOZANO CORVERA DANIEL ANTONIO

GARCIA AGUILAR JOSE ALFREDO

**MAESTRO**

DANIEL ESPARZA SOTO

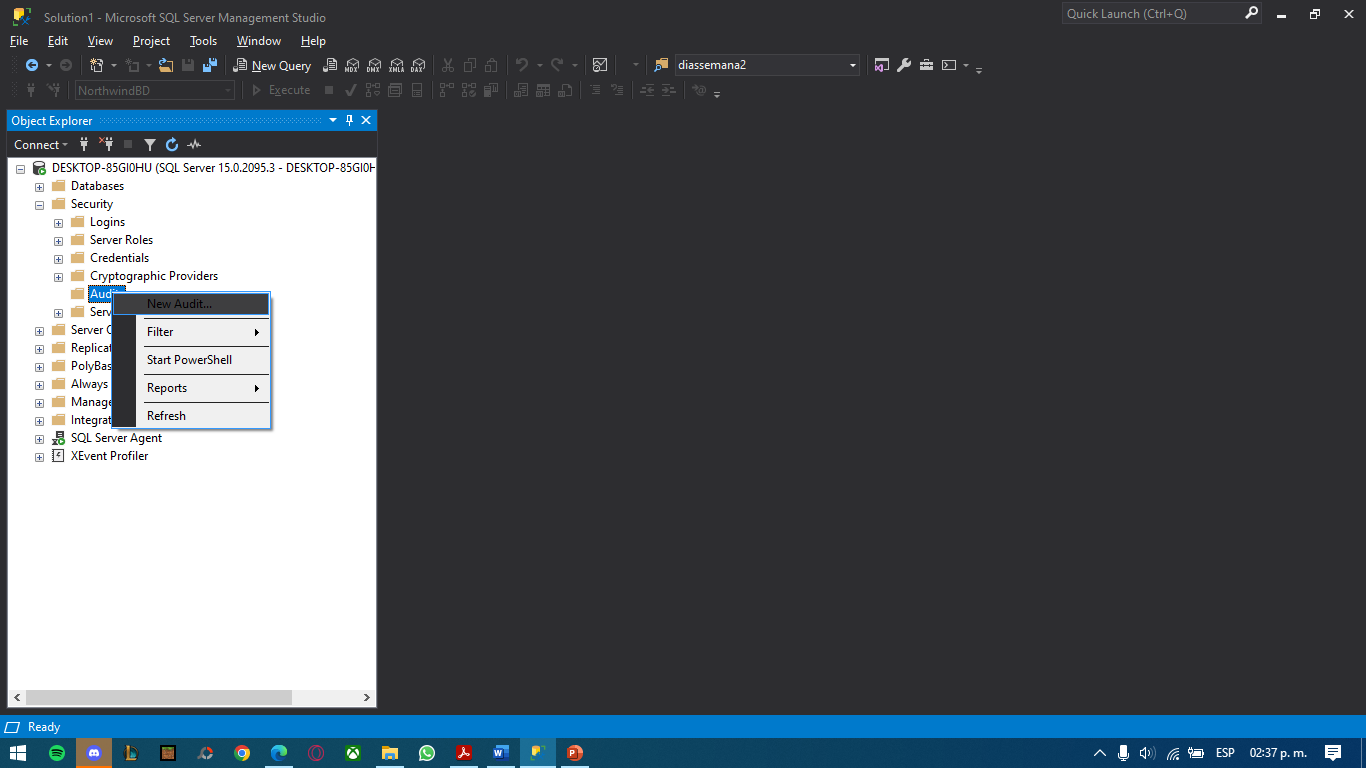
**FECHA**

7-DICIEMBRE-2022

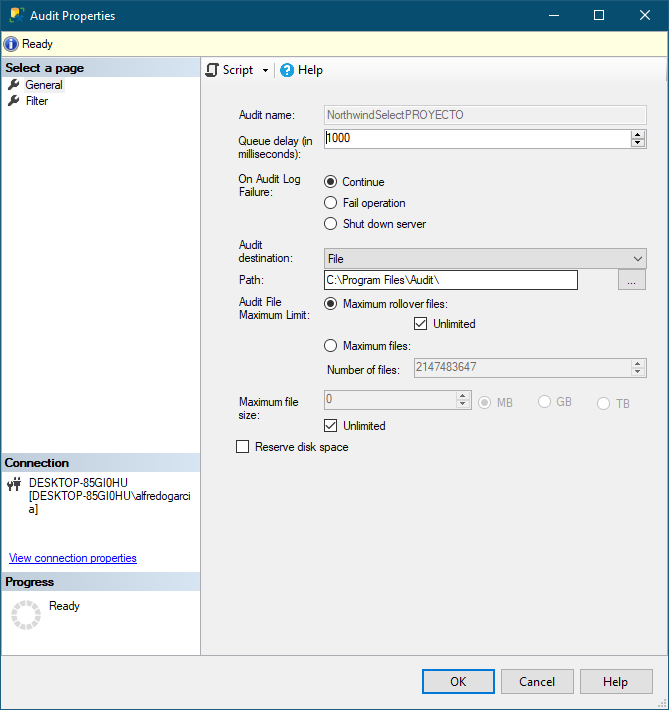
**PROYECTO AUDITORÍAS**

**1.- PROPONER UN MÉTODO DE AUDITAR EL COMANDO SELECT SOBRE TODAS LA TABLAS DE LA BASE DE DATOS NORTHWIND. REALIZAR 5 CONSULTAS Y EXPONGA EL MÉTODO DE REVISIÓN DE LA AUDITORIA.**

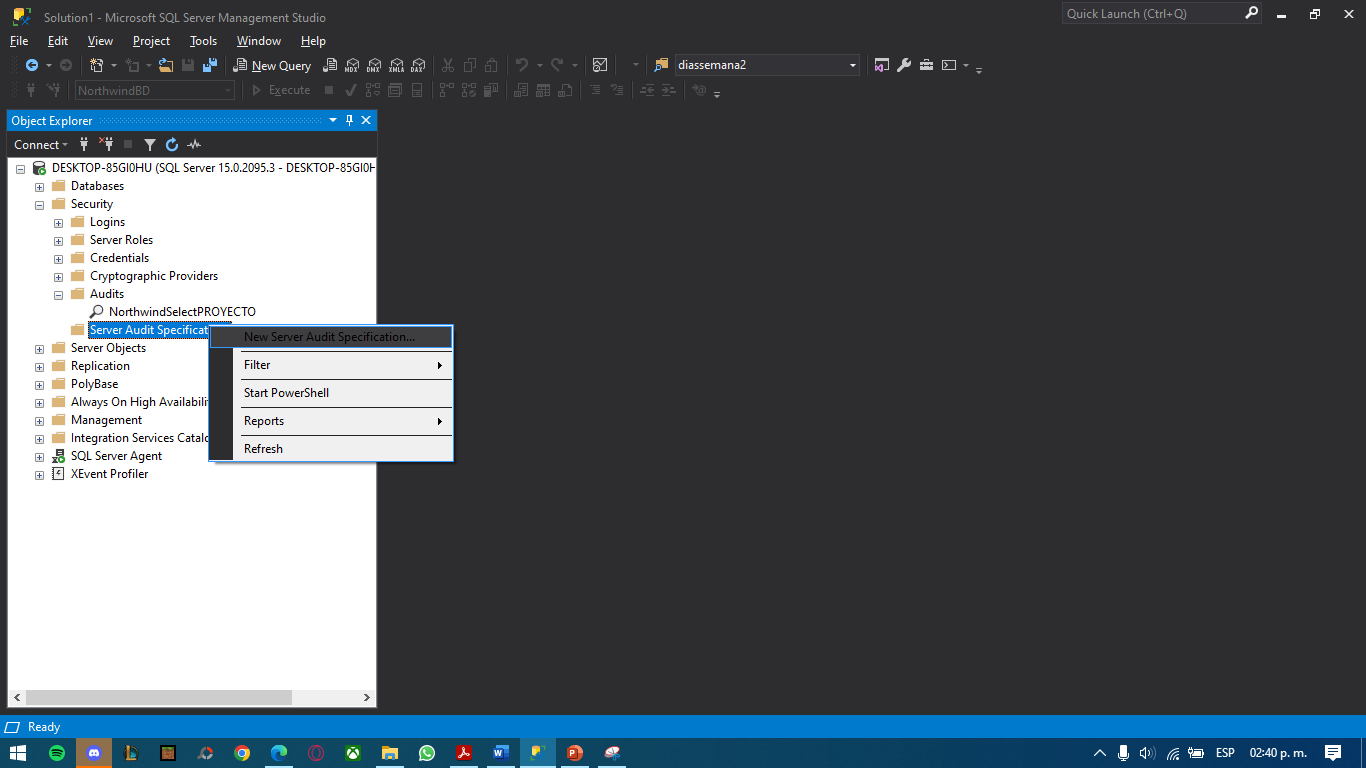
Click derecho y New Audit



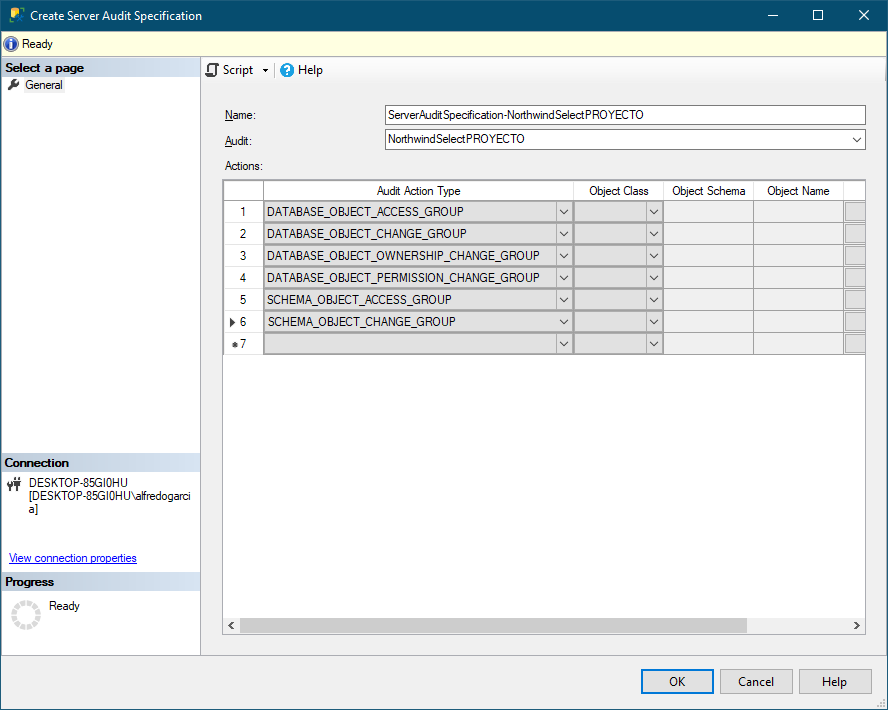
Le damos nombre y la ruta de la carpeta Audit. Ok



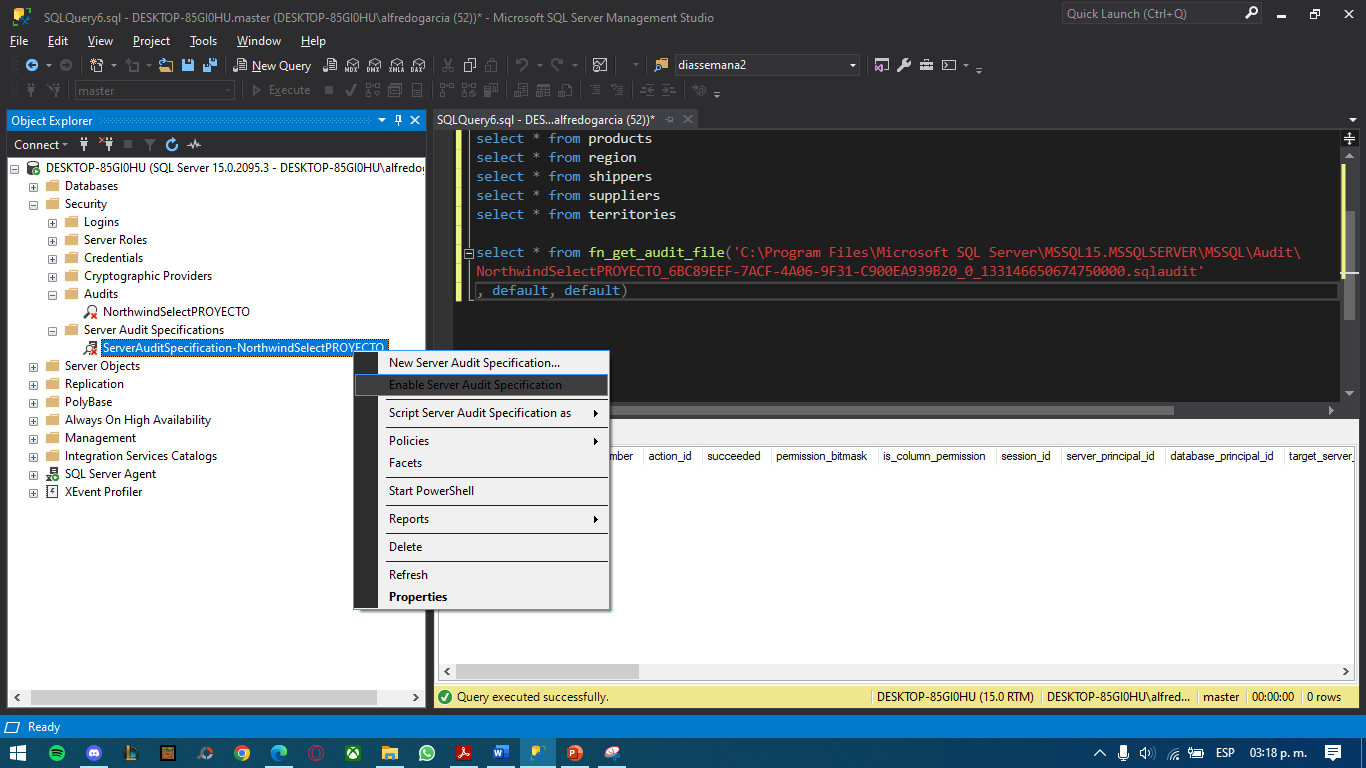
Click derecho en Server Audit Specifications y elegimos New

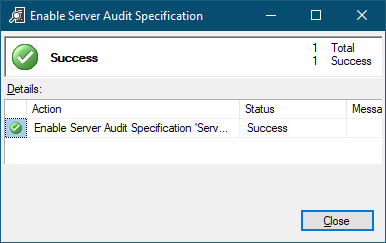


Le damos un nombre, elegimos la NorthwindSelectPROYECTO y la alta de acciones. Ok

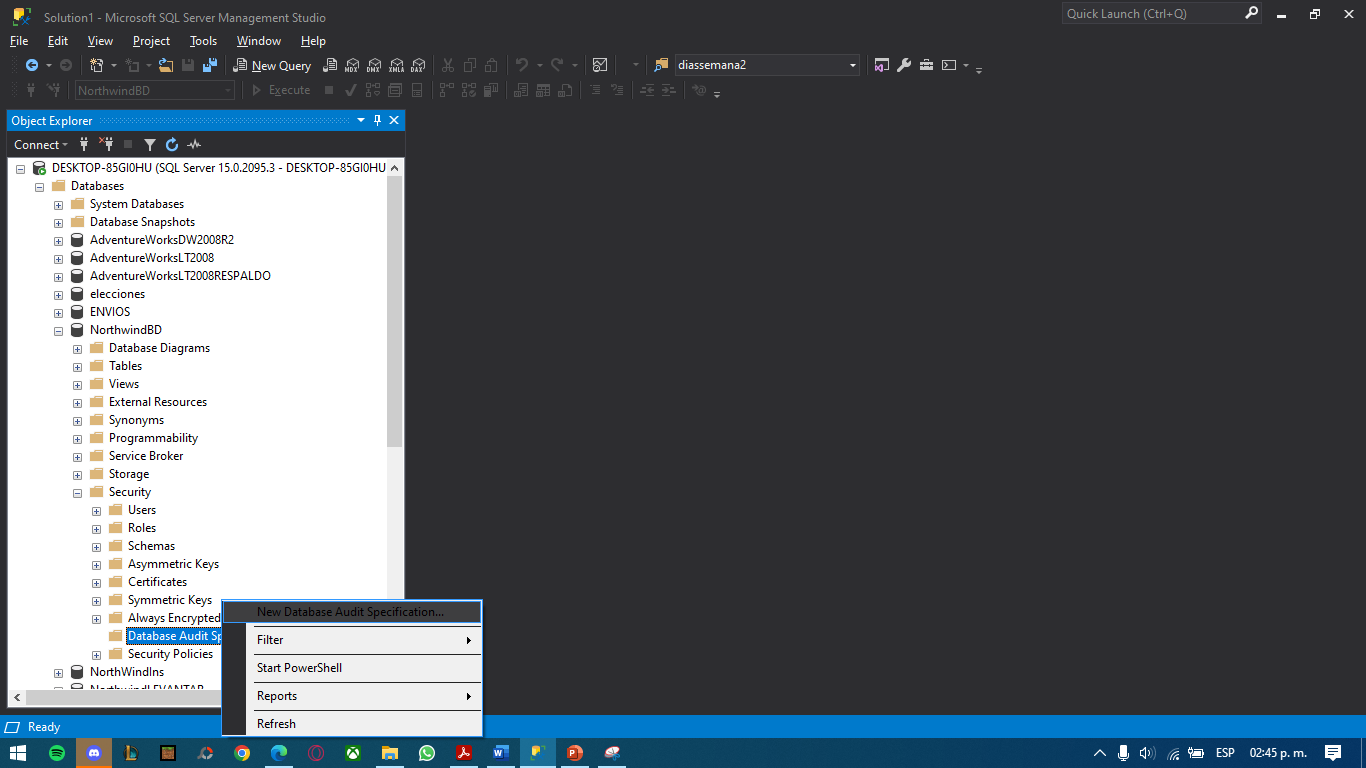


Habilitamos la especificación



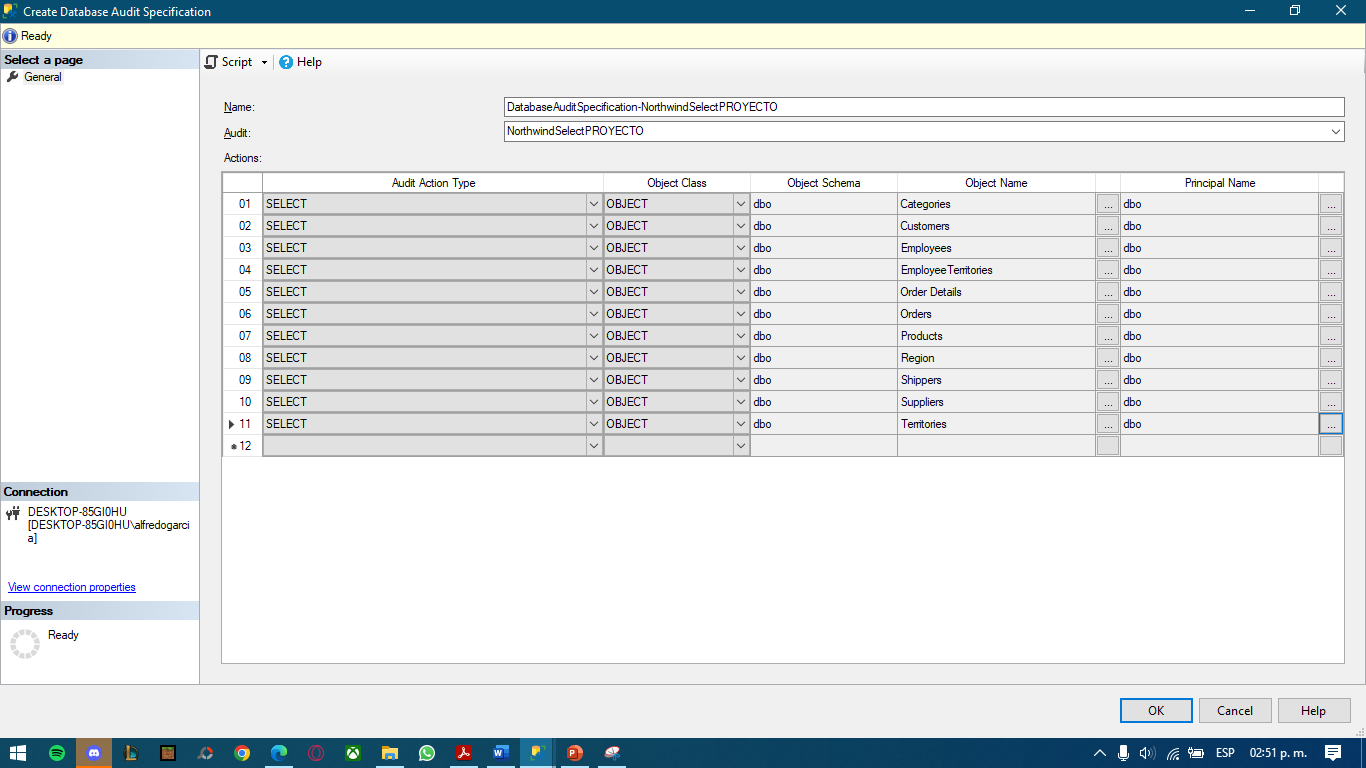


Vamos a la base de datos que ocupamos o sea la Northwind y llegamos hasta Database Audit Specifications, damos New

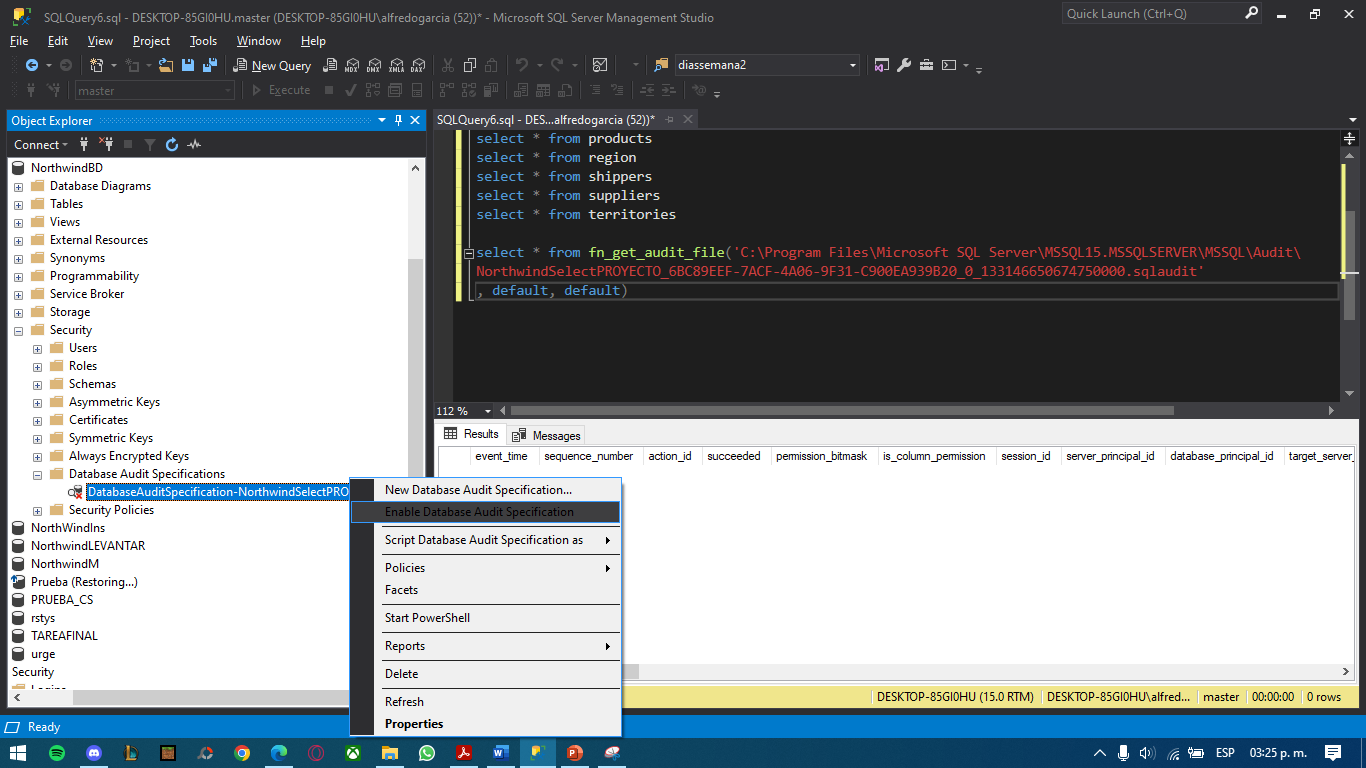


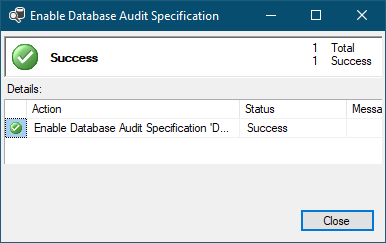
Lo dejamos así:

Ok



Habilitamos la especificación





Hacemos pruebas haciendo todos los selects

select \* from categories

select \* from customers

select \* from employees

select \* from employeeterritories

select \* from [order details]

select \* from orders

select \* from products

select \* from region

select \* from shippers

select \* from suppliers

select \* from territories

select event\_time, statement, object\_name, server\_principal\_name

from fn\_get\_audit\_file('C:\Program Files\Audit\NorthwindSelectPROYECTO\_8C043116-D5D3-4850-BE75-BE1B2CBB6EEF\_0\_133146676869040000.sqlaudit'

, default, default)

where statement like 'select%'

order by event\_time

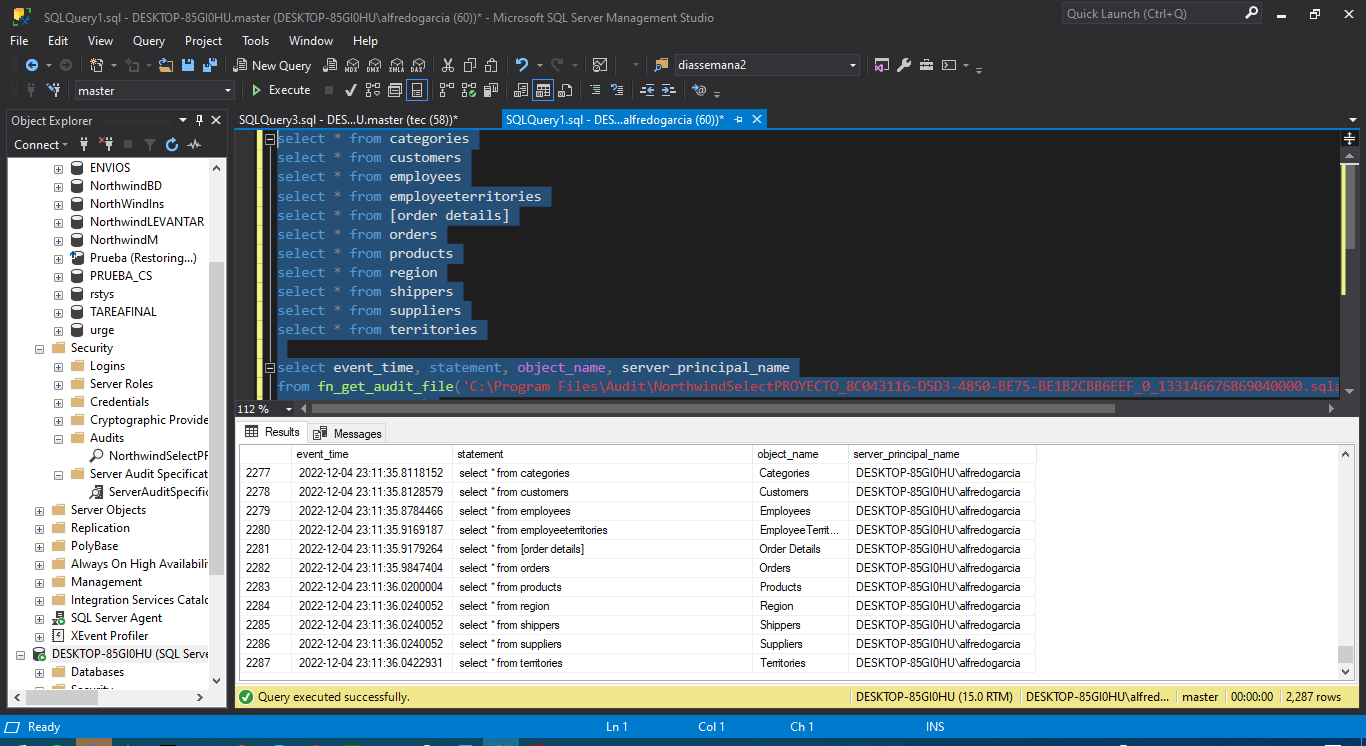
select \*

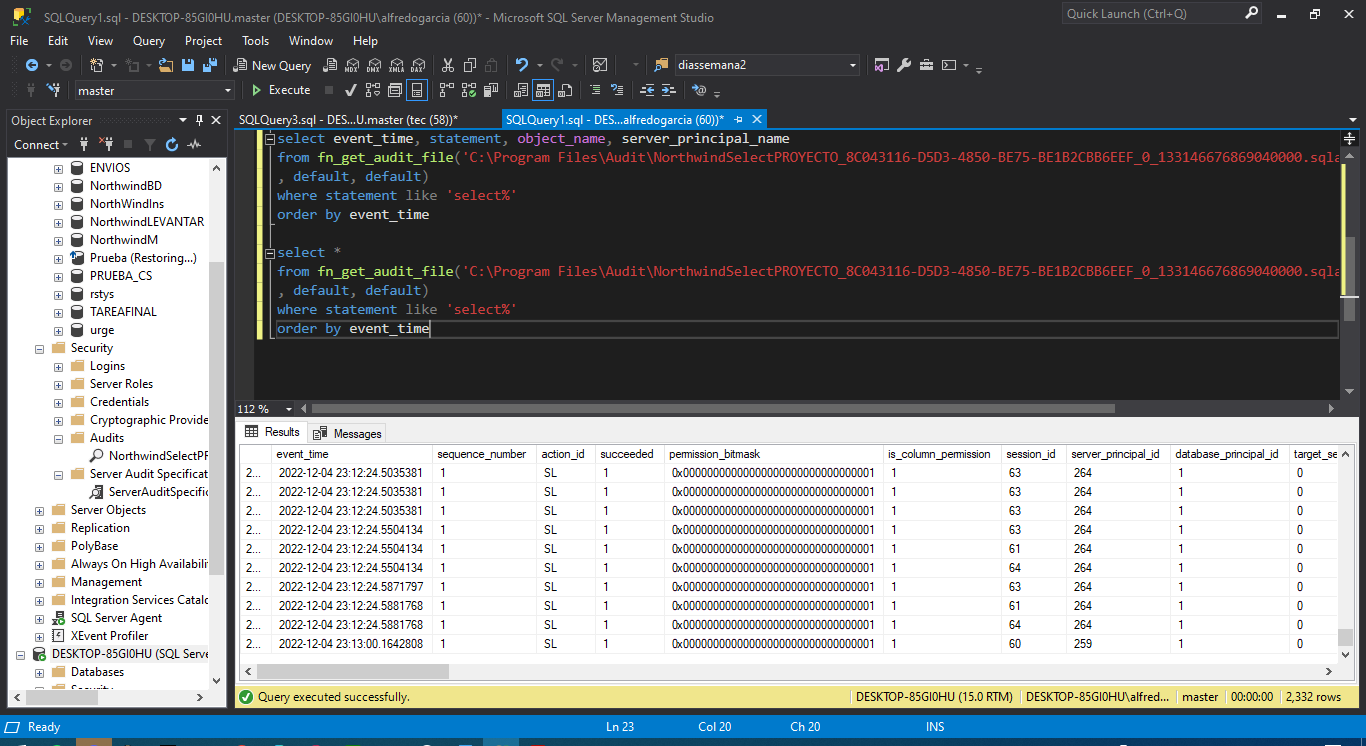
from fn\_get\_audit\_file('C:\Program Files\Audit\NorthwindSelectPROYECTO\_8C043116-D5D3-4850-BE75-BE1B2CBB6EEF\_0\_133146676869040000.sqlaudit'

, default, default)

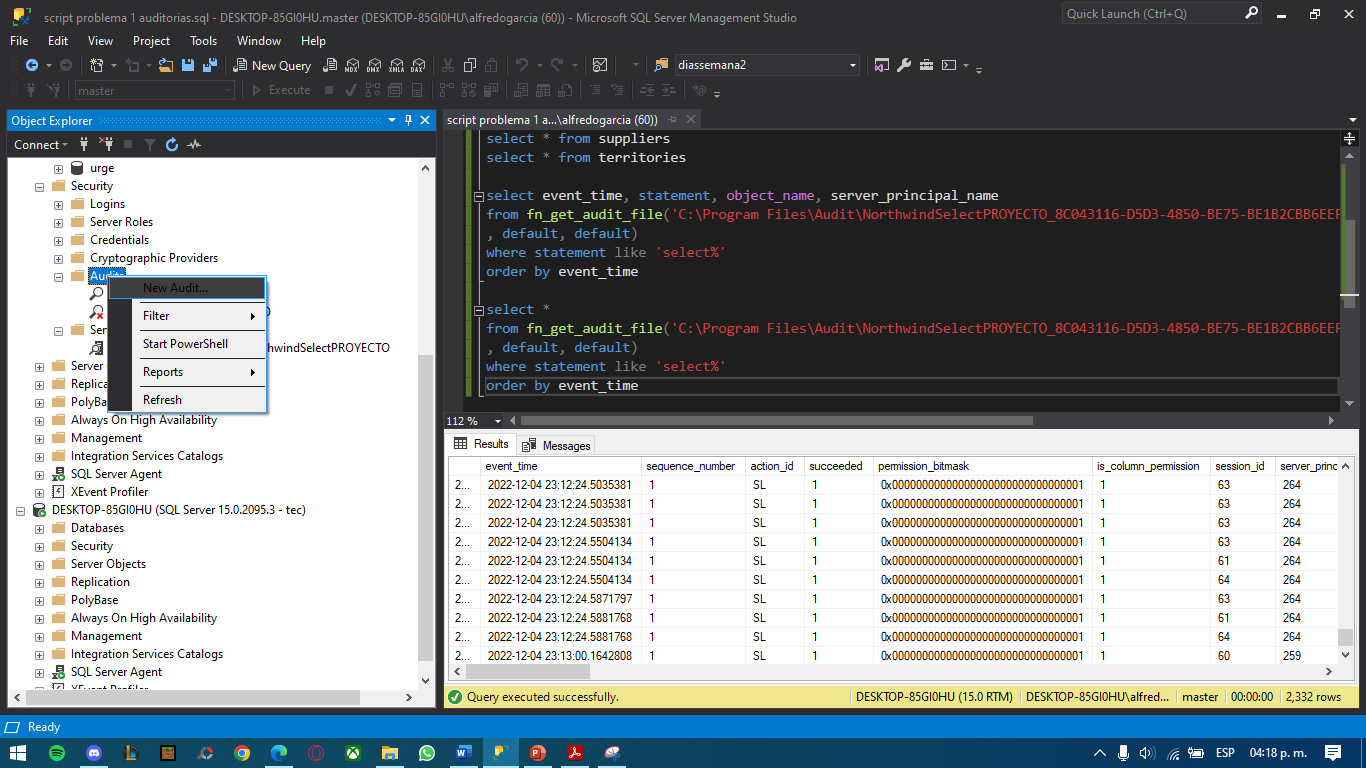
where statement like 'select%'

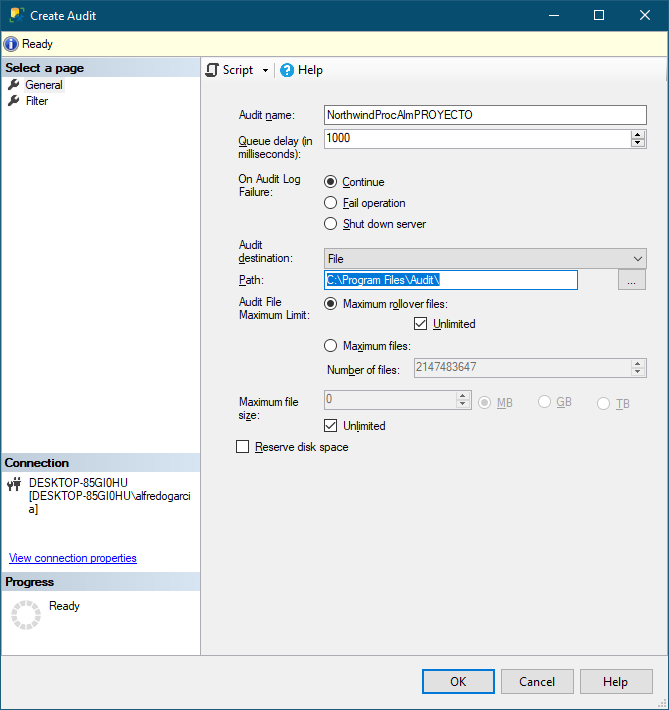
order by event\_time

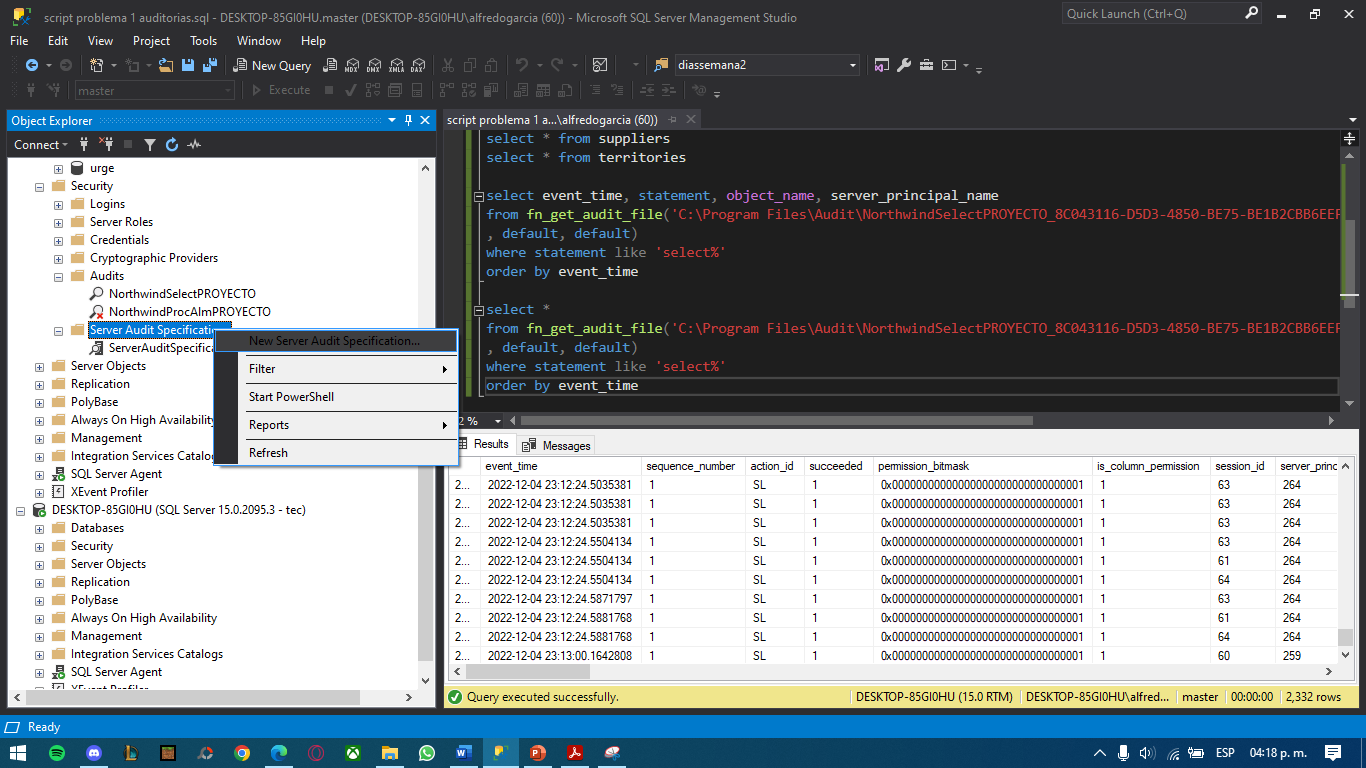


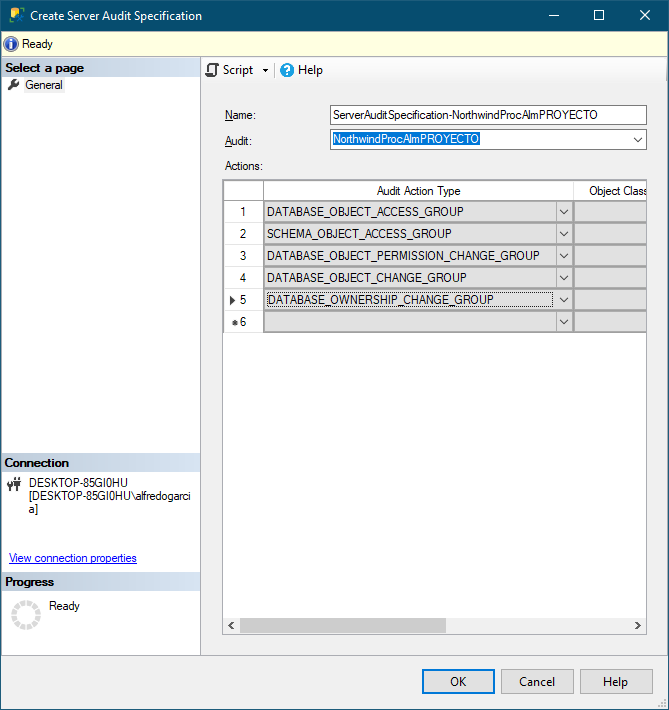


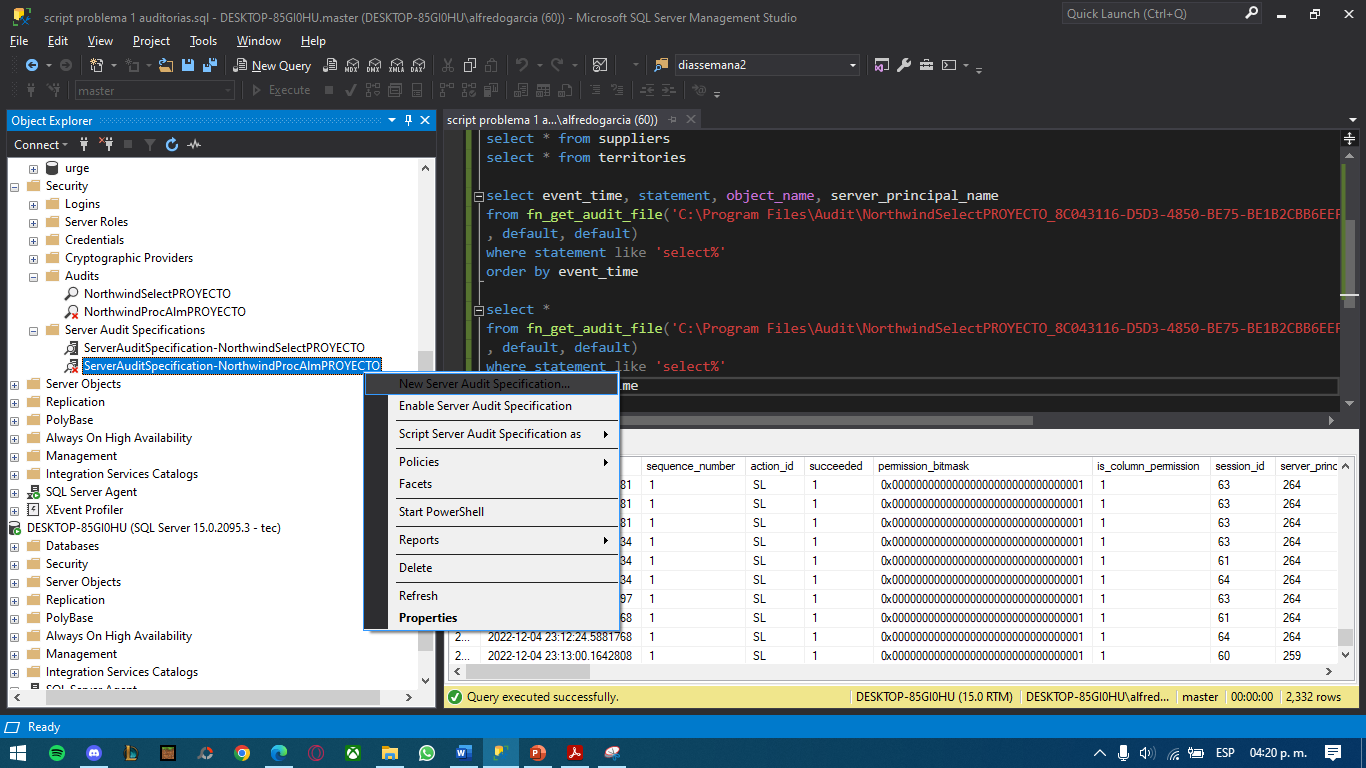
**2.- PROPONER UN MÉTODO PARA AUDITAR LA EXECUCIÓN DE TODOS LOS PROCEDIMIENTOS ALMACENADOS DE LA BD NORTHWIND. CREAR DOS PROCEDIMIENTOS ALMACENADO Y EXPONGA EL MÉTODO DE REVISIÓN DE LA AUDITORÍA.**

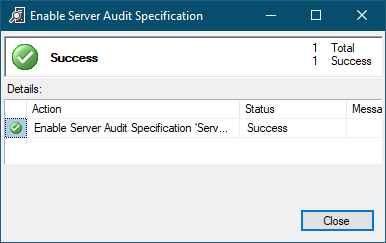


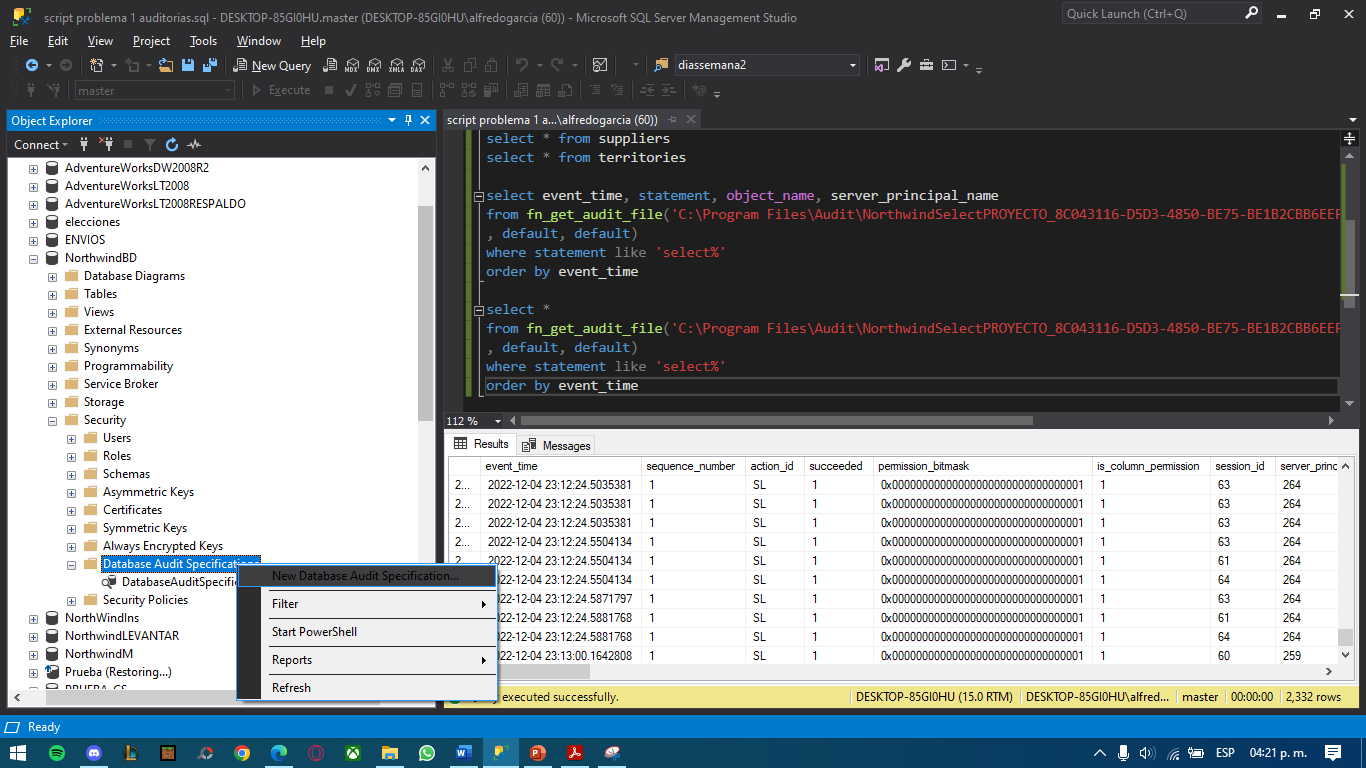












use NorthwindBD

create proc sp\_customers

as

begin

select \* from customers

end

go

execute sp\_customers

--drop procedure sp\_customers

use NorthwindBD

create proc sp\_orders

as

begin

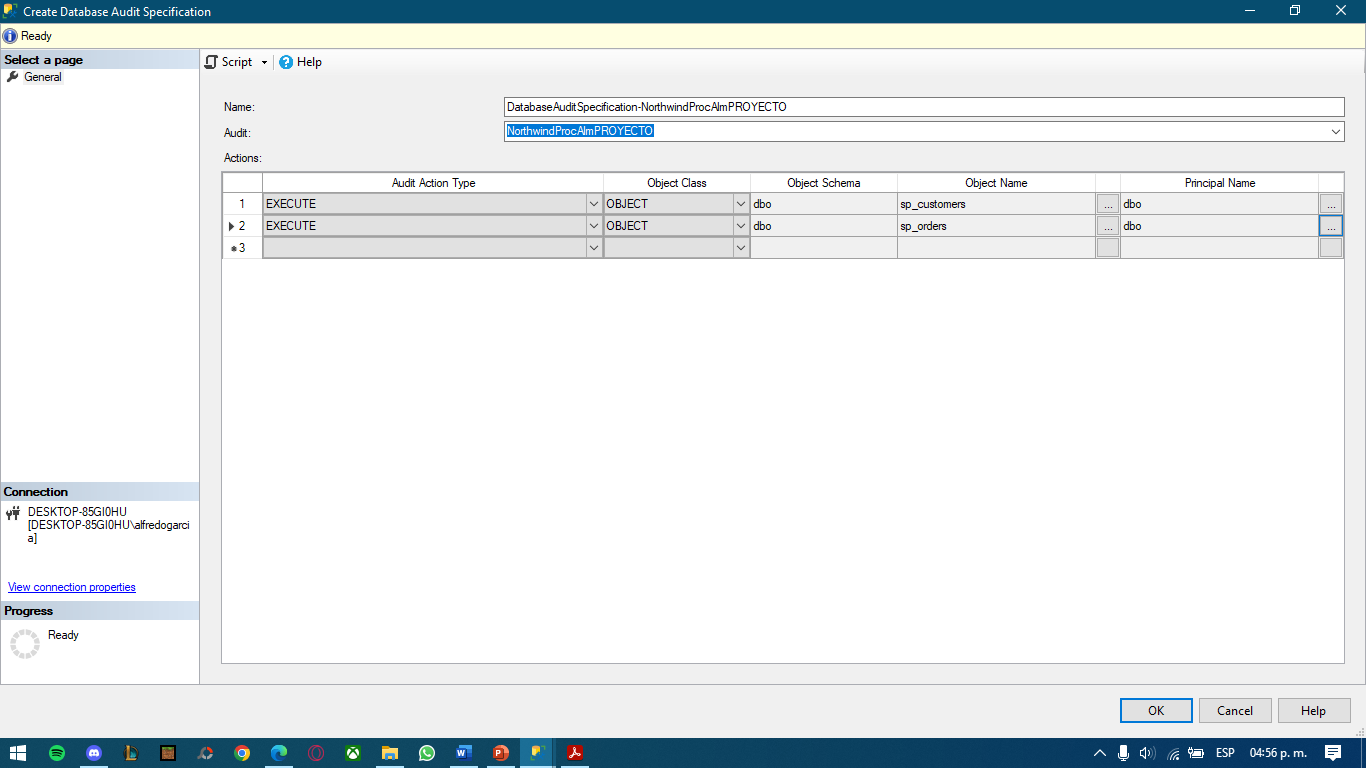
select \* from orders

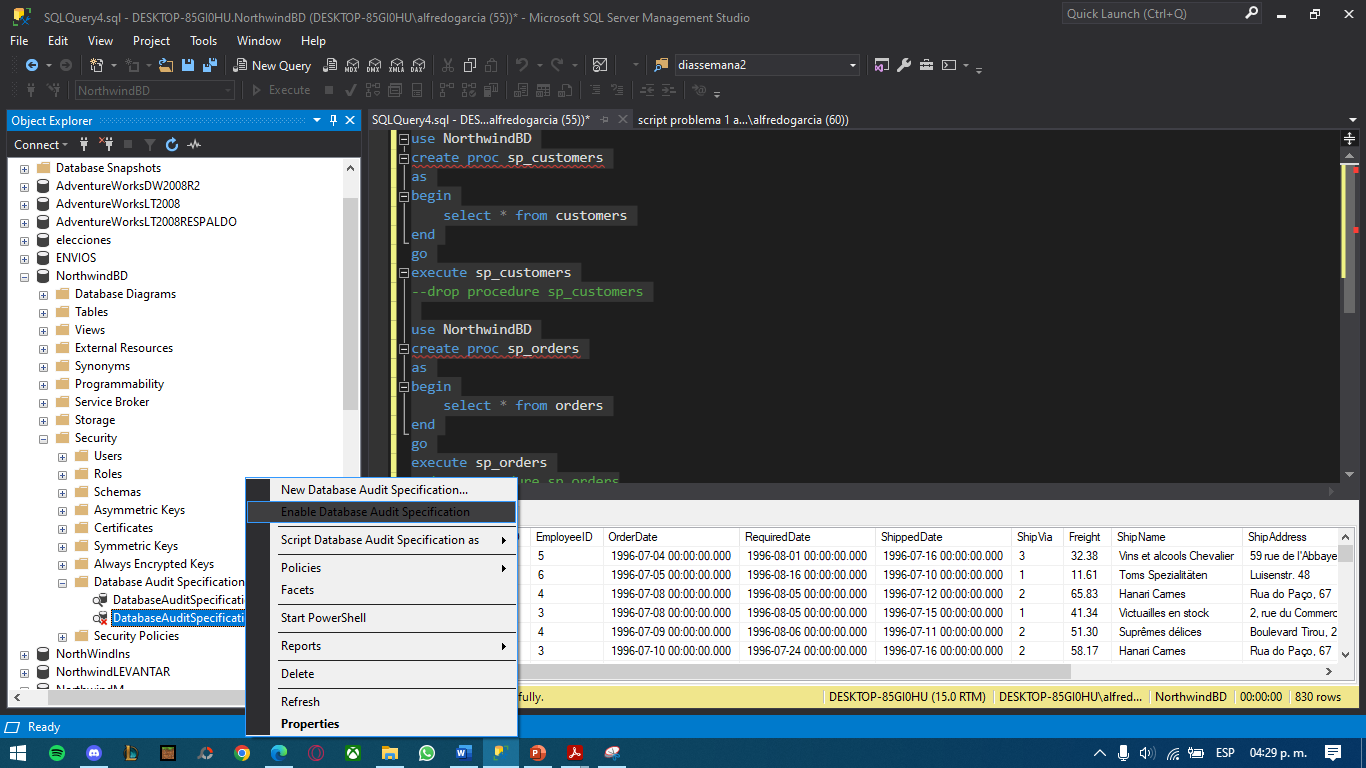
end

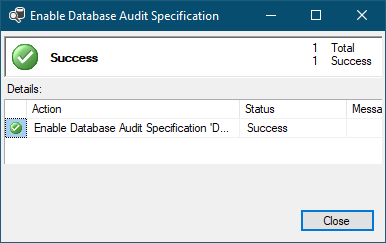
go

execute sp\_orders

--drop procedure sp\_orders







use NorthwindBD

create proc sp\_customers

as

begin

select \* from customers

end

go

execute sp\_customers

--drop procedure sp\_customers

use NorthwindBD

create proc sp\_orders

as

begin

select \* from orders

end

go

execute sp\_orders

--drop procedure sp\_orders

select event\_time, statement, object\_name, server\_principal\_name

from fn\_get\_audit\_file('C:\Program Files\Audit\NorthwindProcAlmPROYECTO\_A4691F73-6362-4036-A92B-230C5A2CC95D\_0\_133146718206340000.sqlaudit',

default, default)

where statement like '%exec%'

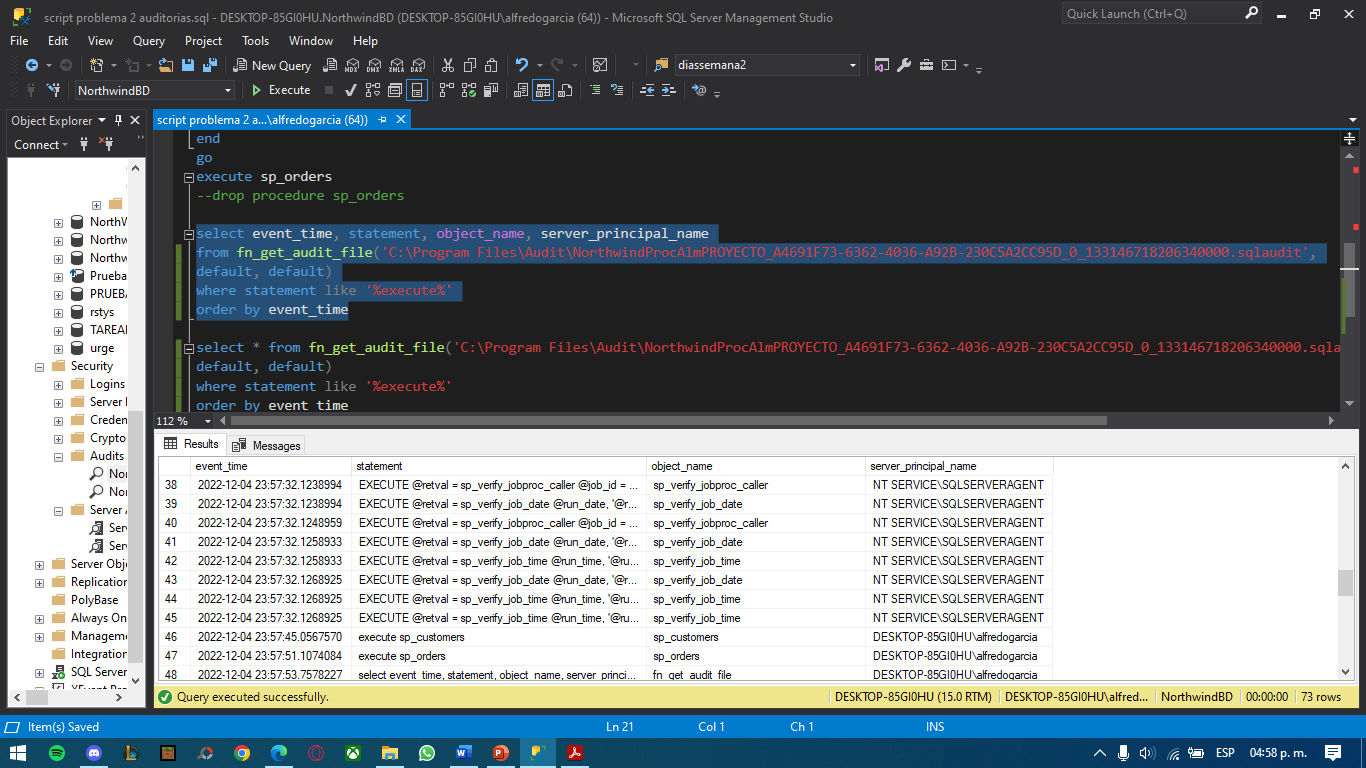
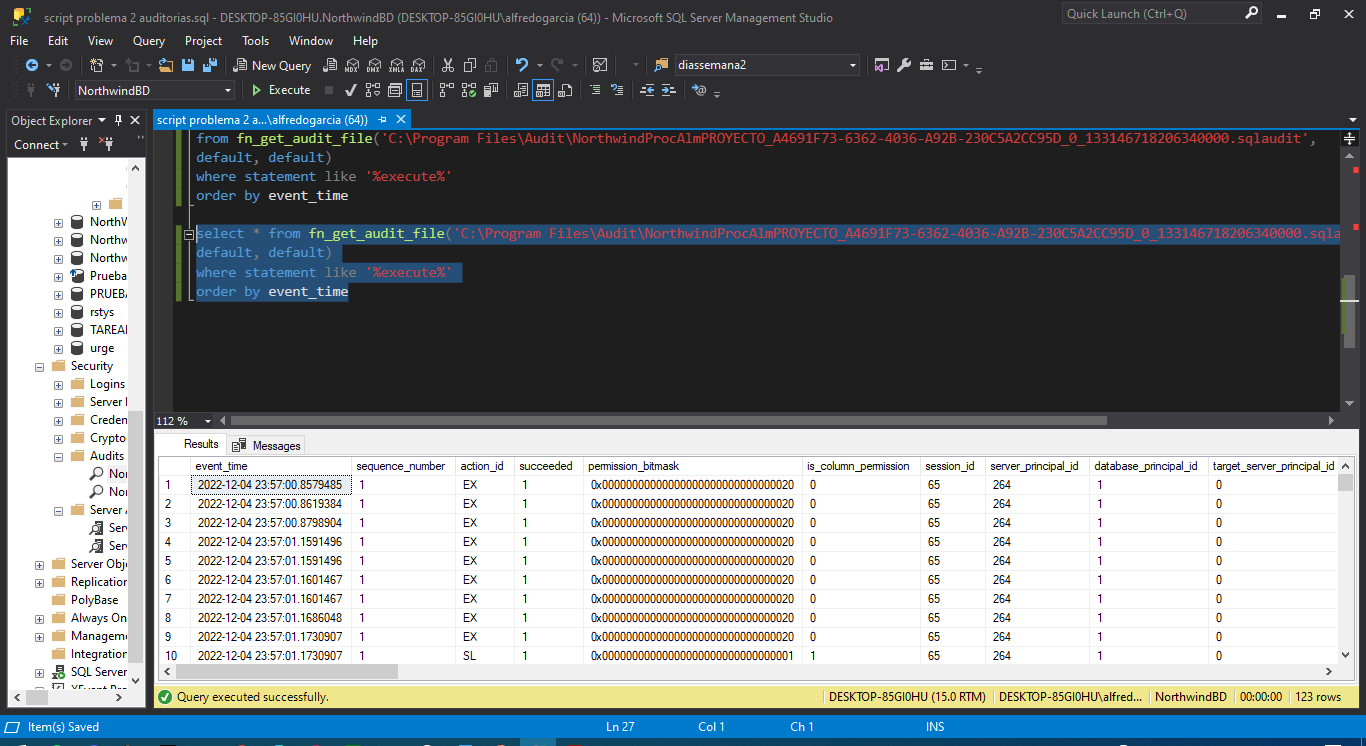
order by event\_time

select \* from fn\_get\_audit\_file('C:\Program Files\Audit\NorthwindProcAlmPROYECTO\_A4691F73-6362-4036-A92B-230C5A2CC95D\_0\_133146718206340000.sqlaudit',

default, default)

where statement like '%exec%'

order by event\_time



**3.- UTILIZANDO TRIGGER DDL A NIVEL SERVIDOR AUDITAR LA CREACIÓN, ELIMINACIÓN O MODIFICACIÓN DE BASE DE DATOS, INICIOS DE SESIÓN Y COPIAS DE SEGURIDAD-RESTAURAIONES. GRABAR EN UNA TABLA LOS SIGUIENTES DATOS:**

**ID**

**FECHA DEL EVENTO,**

**SERVIDOR,-**

**INICIO DE SESIÓN,**

**TIPO DE EVENTO, -**

**TIPO DE OBJETO,-**

**SENTENCIA Y**

**DETALLE DEL EVENTO XML.**

**REALIZAR 5 ACCIONES SOBRE BASE DE DATOS, 5 ACCIONES SOBRE INICIOS DE SESIÓN, 5 COPIAS DE SEGURIDAD Y 5 RESPALDOS.**

use master

create table tabla\_problema3(

ID int identity primary key,

FechaEvento datetime not null,

Servidor varchar(200) not null,

Login varchar(200) not null,

TipoEvento varchar(200) not null,

TipoObjeto varchar(200) null,

Sentencia varchar(max) not null,

DetalleEvento xml not null

);

go

--drop table tabla\_problema3;

create trigger trigger\_problema3 on all server

for CREATE\_DATABASE, DROP\_DATABASE, ALTER\_DATABASE,

--CREATE\_LOGIN, DROP\_LOGIN, ALTER\_LOGIN

DDL\_SERVER\_SECURITY\_EVENTS as insert into tabla\_problema3

select EVENTDATA().value('(/EVENT\_INSTANCE/PostTime)[1]', 'DATETIME') as FechaEvento, EVENTDATA().value('(/EVENT\_INSTANCE/ServerName)[1]', 'VARCHAR(200)') as Servidor,

EVENTDATA().value('(/EVENT\_INSTANCE/LoginName)[1]', 'VARCHAR(200)') as Login, EVENTDATA().value('(/EVENT\_INSTANCE/EventType)[1]', 'VARCHAR(200)') as TipoEvento,

EVENTDATA().value('(/EVENT\_INSTANCE/ObjectType)[1]', 'VARCHAR(200)') as TipoObjeto, EVENTDATA().value('(/EVENT\_INSTANCE/TSQLCommand/CommandText)[1]', 'VARCHAR(200)') as Sentencia,

EVENTDATA() as DetalleEvento

go

--drop trigger trigger\_problema3 on all server

--Acciones Base de datos

use master

create database bdejemplo1

create database bdejemplo2

create database bdejemplo3

drop database bdejemplo1

drop database bdejemplo2

drop database bdejemplo3

--Acciones Inicios de sesión

create login pepito with password = '123'

must\_change , check\_expiration = on

go

create login chuy with password = '123'

must\_change , check\_expiration = on

go

alter login pepito with password = '12345'

go

alter login chuy with password = '12345'

go

sp\_addsrvrolemember pepito, Securityadmin

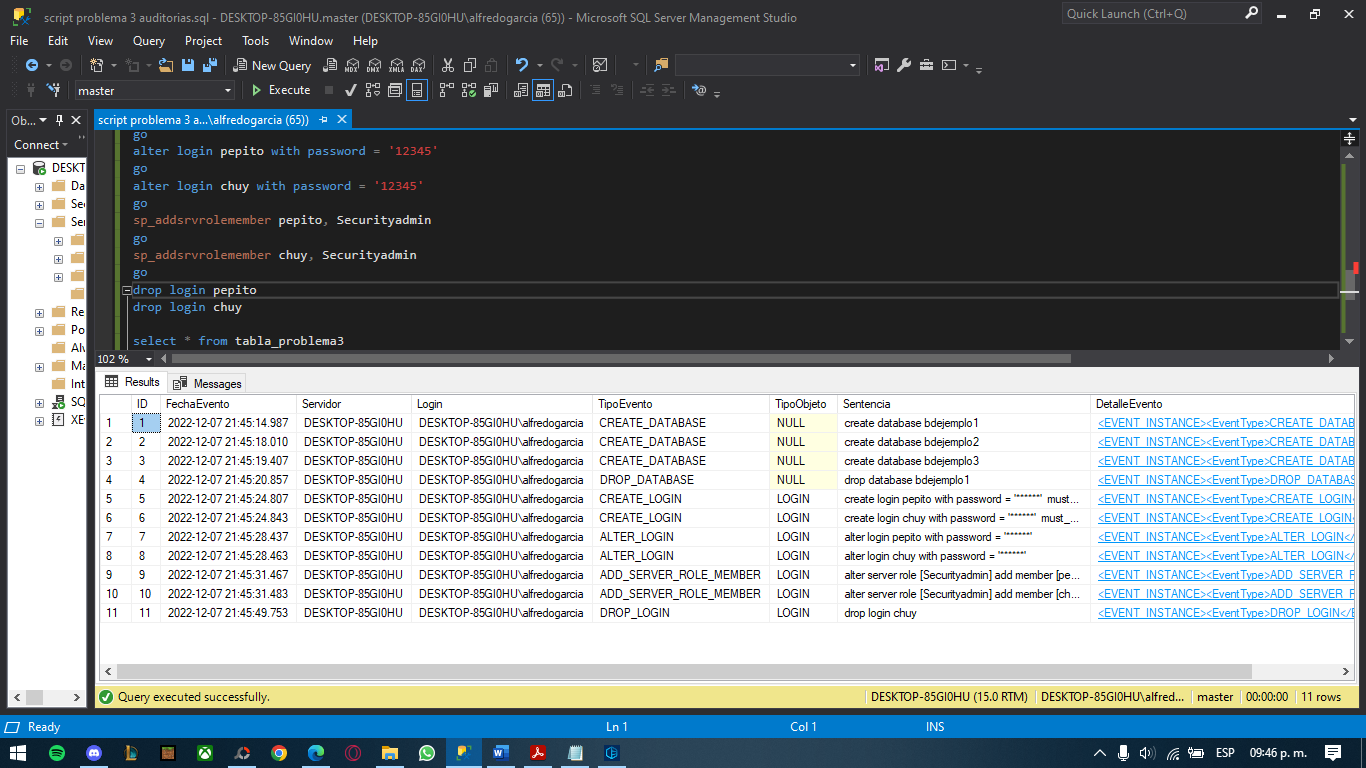
go

sp\_addsrvrolemember chuy, Securityadmin

go

drop login pepito

drop login chuy

select \* from tabla\_problema3

**4.- UTILIZANDO TRIGGER DDL A NIVEL DE BASE DE DATOS (NORTHWIND) AUDITAR LA CREACIÓN, ELIMINACIÓN O MODIFICACIÓN DE TABLAS VISTAS, PROCEDIMIENTOS ALMACENADOS, FUNCIONES Y TRIGGER. GRABAR EN UNA TABLA:**

**CLAVE,**

**FECHA DEL EVENTO,**

**INICIO DE SESIÓN,**

**USUARIO DE BASE DE DATOS,**

**NOMBRE DE BASE DE DATOS,**

**TIPO DE EVENTO,**

**NOMBRE COMPLETO DEL OBJETO: ESQEMA.NOMBREOBJETO,**

**SENTENCIA Y**

**DETALLE DEL EVENTO XML.**

**REALIZAR UN COMANDO PARA VISTAS, PROCEDIMIENTOS ALMACENADOS, FUNCIONES Y TRIGGER.**

use NorthwindBD

go

create table tabla\_problema4 (

ID int identity primary key,

FechaEvento datetime not null,

Login varchar(200) not null,

NombreUsuario varchar(200) not null,

NombreBaseDatos varchar(200) not null,

TipoEvento varchar(200) not null,

NombreObjeto varchar(200) not null,

Sentencia varchar(max) not null,

DetalleEvento xml not null

);

go

--drop table tabla\_problema4

use NorthwindBD

create trigger trigercito on database

for CREATE\_TABLE, DROP\_TABLE, ALTER\_TABLE,

CREATE\_VIEW, DROP\_VIEW, ALTER\_VIEW,

CREATE\_PROCEDURE, DROP\_PROCEDURE, ALTER\_PROCEDURE,

CREATE\_FUNCTION, DROP\_FUNCTION, ALTER\_FUNCTION,

CREATE\_TRIGGER, DROP\_TRIGGER, ALTER\_TRIGGER

--DDL\_VIEW\_EVENTS, DDL\_PROCEDURE\_EVENTS, DDL\_FUNCTION\_EVENTS, DDL\_TRIGGER\_EVENTS

as insert into tabla\_problema4

select EVENTDATA().value('(/EVENT\_INSTANCE/PostTime)[1]', 'DATETIME') as FechaEvento, EVENTDATA().value('(/EVENT\_INSTANCE/LoginName)[1]', 'VARCHAR(200)') as Login,

EVENTDATA().value('(/EVENT\_INSTANCE/UserName)[1]', 'VARCHAR(200)') as NombreUsuario, EVENTDATA().value('(/EVENT\_INSTANCE/DatabaseName)[1]', 'VARCHAR(200)') as NombreBaseDatos,

EVENTDATA().value('(/EVENT\_INSTANCE/EventType)[1]', 'VARCHAR(200)') as TipoEvento, EVENTDATA().value('(/EVENT\_INSTANCE/SchemaName)[1]', 'VARCHAR(200)') + '.' + EVENTDATA().value('(/EVENT\_INSTANCE/ObjectName)[1]', 'VARCHAR(30)') as NombreObjeto,

EVENTDATA().value('(/EVENT\_INSTANCE/TSQLCommand/CommandText)[1]', 'VARCHAR(max)') as Sentencia, EVENTDATA() AS DetalleEvento;

go

--drop trigger trigercito on all server

--Pruebas para Tablas

create table tienda(

tiendaid int,

calle varchar(50)

)

go

alter table tienda add colonia varchar(50)

go

drop table tienda

go

--Pruebas para Vistas

create view vw\_products as

select

p.productid, p.productname, p.quantityperunit, prodnitprice = p.unitprice,

p.unitsinstock, p.unitsonorder, p.reorderlevel,

s.supplierid, s.companyname, s.contactname, s.address,

s.city, s.region, s.postalcode, s.country, s.phone, s.fax, s.homepage,

c.categoryid, c.categoryname, c.description, c.picture

from products p

inner join suppliers s on p.supplierid = s.supplierid

inner join categories c on p.categoryid = c.categoryid

alter view vw\_products as

select \* from products

drop view vw\_products

--Pruebas para Procedimientos Almacenados

use NorthwindBD

create proc sp\_customers2

as

begin

select \* from customers

end

go

alter proc sp\_customers2 as

begin

select \* from [Order Details]

end

go

drop procedure sp\_customers2

use NorthwindBD

create proc sp\_orders2

as

begin

select \* from orders

end

go

alter proc sp\_orders2 as

begin

select \* from Shippers

end

go

drop procedure sp\_orders2

--Pruebas para Funciones

create function fn\_customers()

returns table as

return(

select companyname from customers

)

alter function fn\_customers()

returns table as

return(

select contacttitle from customers

)

drop function fn\_customers

--Pruebas para Triggers

create trigger trigger\_prueba

on employees for delete as

begin

select 'Hey'

end

go

alter trigger trigger\_prueba

on employees for delete as

begin

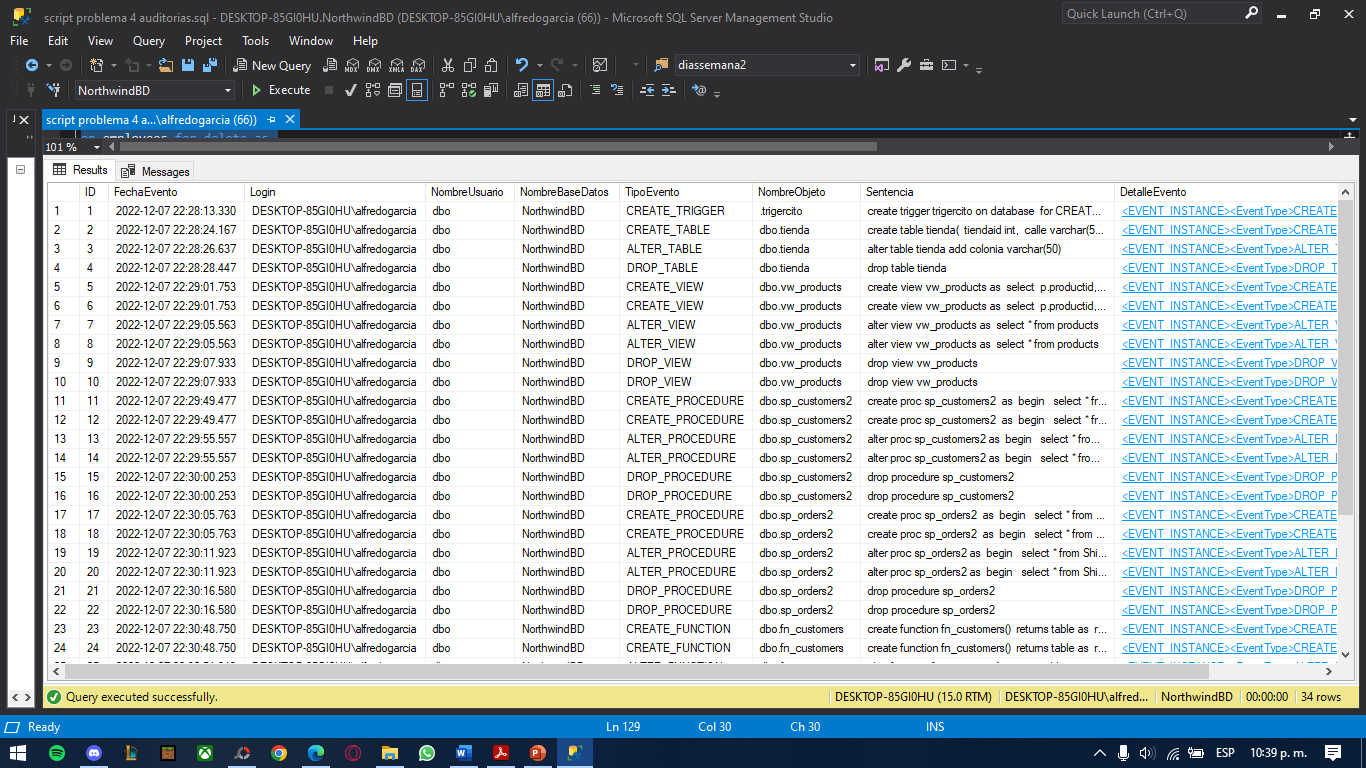
select ':D'

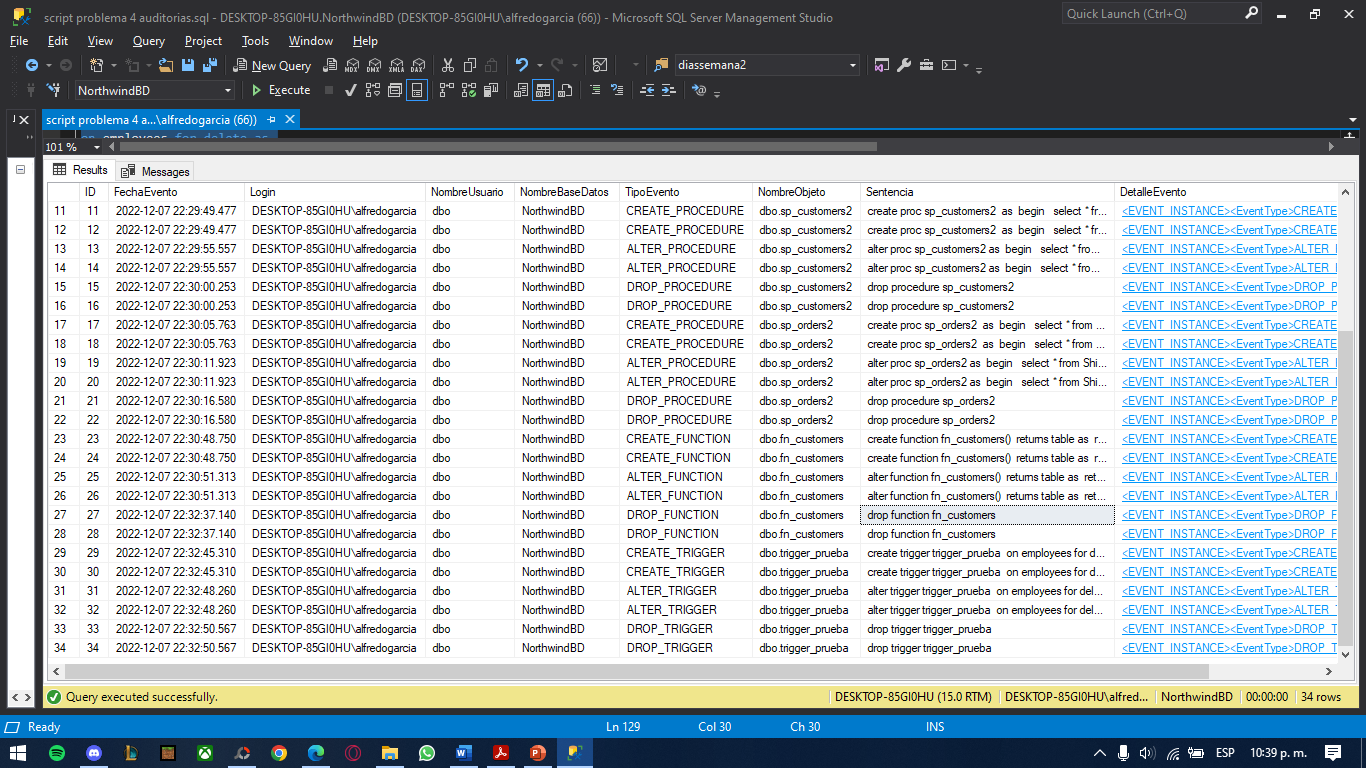
end

go

drop trigger trigger\_prueba

select \* from tabla\_problema4





**5.- AUDITAR LOS CAMBIOS DE LAS TABLAS DE LA BASE DE DATOS NORTHWIND, GRABAR EN UNA TABLA LOS CAMPOS:**

**TIPOTRANSACCION: INSERT, UPDATE, DELETE**

**NOMBRE DE LA TABLA**

**CAMPO DE LA LLAVE PRIMARIA**

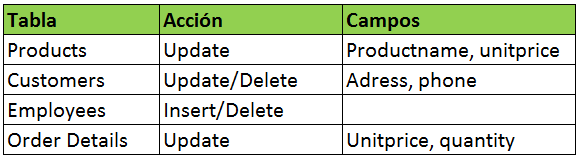
**CAMPO MODIFICADO**

**VALOR ORIGINAL**

**VALOR NUEVO**

**FECHA DE LA TRANSACCIÓN**

**NOMBRE DE USUARIO**



use NorthwindBD

create table tabla\_problema5 (

TipoTransaccion varchar(1000),

NombreTabla varchar(1000),

CampoPK varchar(1000),

CampoModificado varchar(1000),

ValorOriginal Varchar(1000),

ValorNuevo Varchar(1000),

FechaTransaccion DateTime Not null constraint DF\_TipoCambios\_FechaTran DEFAULT (getdate()),

NombreUsuario varchar(1000)

)

--drop table tabla\_problema5

--Products: Accion = Update, Campos = ProductName, UnitPrice

create trigger tr\_products ON products for update

as begin

declare @TipoTransaccion varchar(1000)

declare @NombreTabla varchar(1000)

declare @CampoPK varchar(1000)

declare @CampoModificado varchar(1000)

declare @ValorOriginal Varchar(1000)

declare @ValorNuevo Varchar(1000)

declare @FechaTransaccion datetime

declare @NombreUsuario varchar(1000)

if exists(select \* from inserted)

if exists(select \* from deleted)

select @TipoTransaccion = 'Update'

else

select @TipoTransaccion = 'Insert'

else

select @TipoTransaccion = 'Delete'

select @NombreTabla = 'Products'

select @NombreUsuario = SYSTEM\_USER

select @FechaTransaccion = GETDATE()

IF UPDATE(ProductName) --validar si se cambia el campo ProductName

begin

SELECT @CampoModificado = 'ProductName'

SELECT @ValorNuevo = CONVERT(varchar(1000),ProductName) FROM inserted

SELECT @ValorOriginal = CONVERT(varchar(1000),ProductName) FROM deleted

end

ELSE IF UPDATE(UnitPrice) --validar si se cambia el campo UnitPrice

begin

SELECT @CampoModificado = 'UnitPrice'

SELECT @ValorNuevo = CONVERT(varchar(1000),UnitPrice) FROM inserted

SELECT @ValorOriginal = CONVERT(varchar(1000),UnitPrice) FROM deleted

end

if exists (SELECT \* FROM inserted)

SELECT @CampoPK = 'ProductID=' + convert(varchar(1000),ProductID) FROM inserted

else

SELECT @CampoPK = 'ProductID=' + convert(varchar(1000),ProductID) FROM deleted

INSERT tabla\_problema5(TipoTransaccion,NombreTabla,CampoPK,CampoModificado,ValorOriginal,ValorNuevo,FechaTransaccion,NombreUsuario)

VALUES (@TipoTransaccion,@NombreTabla,@CampoPK,@CampoModificado,@ValorOriginal,@ValorNuevo,@FechaTransaccion,@NombreUsuario)

END

GO

--drop trigger tr\_products

--update

update products set ProductName = 'Lechuga' where productid = '1'

update products set UnitPrice = '35' where productid = '1'

select \* from products

select \* from tabla\_problema5

go

--Customers: Accion = Update/Delete, Campos = Address, Phone

create trigger tr\_customers ON customers for update, delete

as begin

declare @TipoTransaccion varchar(1000)

declare @NombreTabla varchar(1000)

declare @CampoPK varchar(1000)

declare @CampoModificado varchar(1000)

declare @ValorOriginal Varchar(1000)

declare @ValorNuevo Varchar(1000)

declare @FechaTransaccion datetime

declare @NombreUsuario varchar(1000)

if exists(select \* from inserted)

if exists(select \* from deleted)

select @TipoTransaccion = 'Update'

else

select @TipoTransaccion = 'Insert'

else

select @TipoTransaccion = 'Delete'

select @NombreTabla = 'Customers'

select @NombreUsuario = SYSTEM\_USER

select @FechaTransaccion = GETDATE()

IF UPDATE(Address) --validar si se cambia el campo Address

begin

SELECT @CampoModificado = 'Address'

SELECT @ValorNuevo = CONVERT(varchar(1000),Address) FROM inserted

SELECT @ValorOriginal = CONVERT(varchar(1000),Address) FROM deleted

end

ELSE IF UPDATE(Phone) --validar si se cambia el campo Phone

begin

SELECT @CampoModificado = 'Phone'

SELECT @ValorNuevo = CONVERT(varchar(1000),Phone) FROM inserted

SELECT @ValorOriginal = CONVERT(varchar(1000),Phone) FROM deleted

end

if exists (SELECT \* FROM inserted)

SELECT @CampoPK = 'CustomerID=' + convert(varchar(1000),CustomerID) FROM inserted

else

SELECT @CampoPK = 'CustomerID=' + convert(varchar(1000),CustomerID) FROM deleted

INSERT tabla\_problema5(TipoTransaccion,NombreTabla,CampoPK,CampoModificado,ValorOriginal,ValorNuevo,FechaTransaccion,NombreUsuario)

VALUES (@TipoTransaccion,@NombreTabla,@CampoPK,@CampoModificado,@ValorOriginal,@ValorNuevo,@FechaTransaccion,@NombreUsuario)

END

GO

--drop trigger tr\_customers

--update

update customers set Address = 'Las Quintas' where CustomerID = 'AROUT'

update customers set phone = '6671230064' where CustomerID = 'AROUT'

--un insert para hacer el delete

INSERT "Customers" VALUES('ZZZZZ','Tradição Hipermercados','Anabela Domingues','Sales Representative','Av. Inês de Castro, 414','Sao Paulo','SP','05634-030','Brazil','(11) 555-2167','(11) 555-2168')

delete customers where CustomerID like 'ZZZZZ'

select \* from customers

select \* from tabla\_problema5

go

--Order Details: Accion = Update, Campos = UnitPrice, Quantity

create trigger tr\_orderdetails ON [Order Details] for update

as begin

declare @TipoTransaccion varchar(1000)

declare @NombreTabla varchar(1000)

declare @CampoPK varchar(1000)

declare @CampoModificado varchar(1000)

declare @ValorOriginal Varchar(1000)

declare @ValorNuevo Varchar(1000)

declare @FechaTransaccion datetime

declare @NombreUsuario varchar(1000)

if exists(select \* from inserted)

if exists(select \* from deleted)

select @TipoTransaccion = 'Update'

else

select @TipoTransaccion = 'Insert'

else

select @TipoTransaccion = 'Delete'

select @NombreTabla = 'Order Details'

select @NombreUsuario = SYSTEM\_USER

select @FechaTransaccion = GETDATE()

IF UPDATE(UnitPrice) --validar si se cambia el campo UnitPrice

begin

SELECT @CampoModificado = 'UnitPrice'

SELECT @ValorNuevo = CONVERT(varchar(1000),UnitPrice) FROM inserted

SELECT @ValorOriginal = CONVERT(varchar(1000),UnitPrice) FROM deleted

end

ELSE IF UPDATE(Quantity) --validar si se cambia el campo Quantity

begin

SELECT @CampoModificado = 'Quantity'

SELECT @ValorNuevo = CONVERT(varchar(1000),Quantity) FROM inserted

SELECT @ValorOriginal = CONVERT(varchar(1000),Quantity) FROM deleted

end

if exists (SELECT \* FROM inserted)

SELECT @CampoPK = 'OrderID=' + convert(varchar(1000),OrderID) FROM inserted

else

SELECT @CampoPK = 'OrderID=' + convert(varchar(1000),OrderID) FROM deleted

INSERT tabla\_problema5(TipoTransaccion,NombreTabla,CampoPK,CampoModificado,ValorOriginal,ValorNuevo,FechaTransaccion,NombreUsuario)

VALUES (@TipoTransaccion,@NombreTabla,@CampoPK,@CampoModificado,@ValorOriginal,@ValorNuevo,@FechaTransaccion,@NombreUsuario)

END

GO

--drop trigger tr\_orderdetails

--un insert para hacer el update

INSERT "Order Details" VALUES(10251,53,32.8,25,0) --el 10249 ya existe en la tabla orders

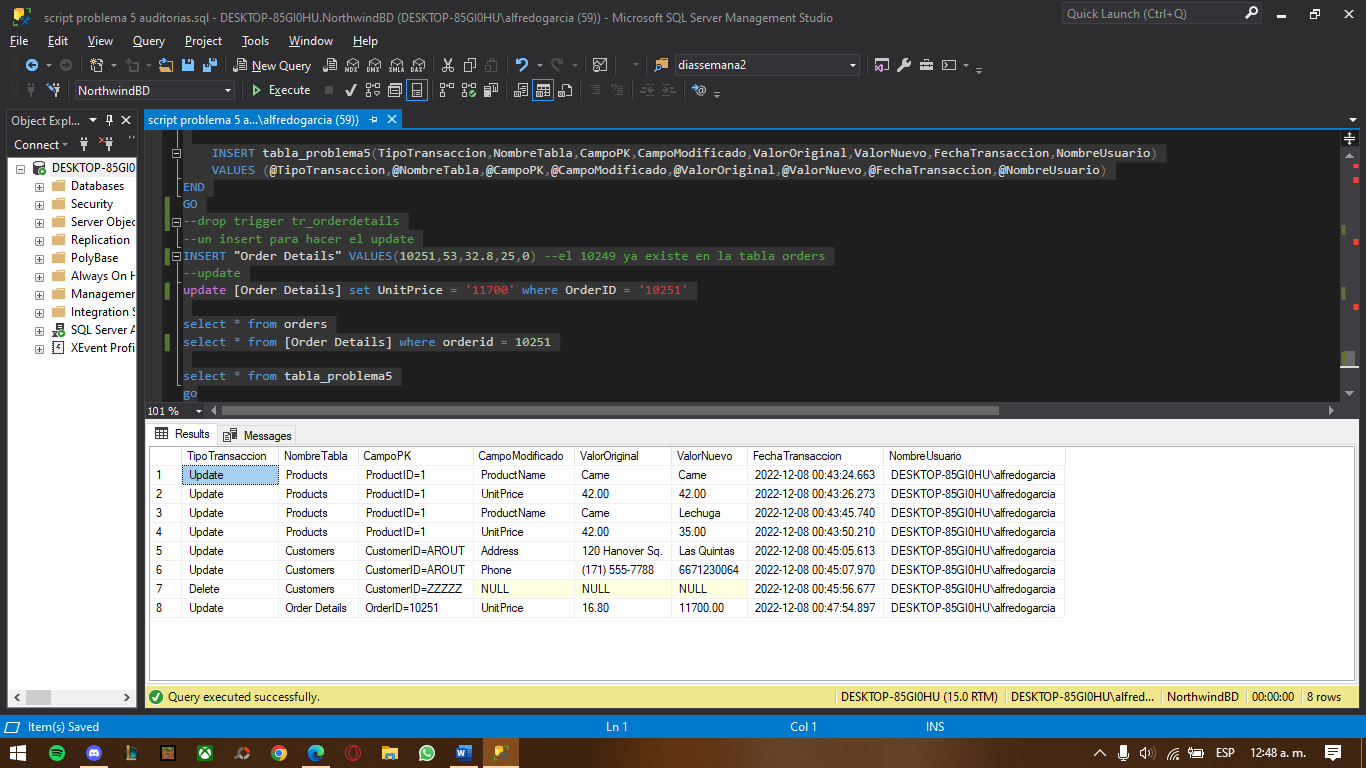
--update

update [Order Details] set UnitPrice = '11700' where OrderID = '10251'

select \* from orders

select \* from [Order Details] where orderid = 10251

select \* from tabla\_problema5

go