Contenido

[IMPLEMENTAR CARGA DIM EMPLEADO 3](#_Toc142000528)

[**MODELO COPO DE NIEVE** 4](#_Toc142000529)

[ORIGEN 1 STAGE DIM EMPLEADO 4](#_Toc142000530)

[EDITOR AVANZADO 5](#_Toc142000531)

[COMPONENTE DE SCRIPT 5](#_Toc142000532)

[ORIGEN 2 DATAMART\_NORTHWND DIM EMPLEADO 7](#_Toc142000533)

[EDITOR AVANZADO 7](#_Toc142000534)

[COMPONENTE DE SCRIPT 7](#_Toc142000535)

[COMBINACION DE MEZCLA 8](#_Toc142000536)

[Insertar en Metadatos Pre-Execute 12](#_Toc142000537)

[Insertar en Metadatos Post-Execute 13](#_Toc142000538)

[IMPLEMENTAR CARGA DIM CATEGORIA 16](#_Toc142000539)

[Stage Categoria 16](#_Toc142000540)

[Data Mark Categoría 16](#_Toc142000541)

[En transformación de Stage MD5 17](#_Toc142000542)

[En transformación de Data-Mart MD5 19](#_Toc142000543)

[Combinación de mezcla Stage izquierda y Data-Mart Derecha 20](#_Toc142000544)

[División Condicional 20](#_Toc142000545)

[Contar Filas 20](#_Toc142000546)

[Destino Data – Mart Categoría (nuevos) 21](#_Toc142000547)

[Comando OLDB Insertar Registros Modificados 21](#_Toc142000548)

[Comando OLDB Actualizar Registros Modificados 22](#_Toc142000549)

[Insertar en Metadatos Pre-Execute 23](#_Toc142000550)

[Insertar en Metadatos Post-Execute 25](#_Toc142000551)

[IMPLEMENTAR CARGA DIM PRODUCTO 26](#_Toc142000552)

[Stage Producto 26](#_Toc142000553)

[Data Mark Producto 27](#_Toc142000554)

[En transformación de Stage MD5 28](#_Toc142000555)

[En transformación de Data-Mart MD5 30](#_Toc142000556)

[Combinación de mezcla Stage izquierda y Data-Mart Derecha 31](#_Toc142000557)

[División Condicional 31](#_Toc142000558)

[Contar Filas 31](#_Toc142000559)

[Destino Data – Mart Categoría (nuevos) 32](#_Toc142000560)

[Comando OLDB Insertar Registros Modificados 32](#_Toc142000561)

[Comando OLDB Actualizar Registros Modificados 33](#_Toc142000562)

[Agregar Búsqueda Nuevo 35](#_Toc142000563)

[Insertar en Metadatos Pre-Execute 36](#_Toc142000564)

[Insertar en Metadatos Post-Execute 37](#_Toc142000565)

[IMPLEMENTAR CARGA DIM CLIENTE 39](#_Toc142000566)

[Stage Cliente 39](#_Toc142000567)

[Data Mark Cliente 39](#_Toc142000568)

[En transformación de Stage MD5 39](#_Toc142000569)

[Combinación de mezcla Stage izquierda y Data-Mart Derecha 40](#_Toc142000570)

[División Condicional 40](#_Toc142000571)

[Contar Filas 40](#_Toc142000572)

[Destino Data – Mart Cliente (nuevos) 41](#_Toc142000573)

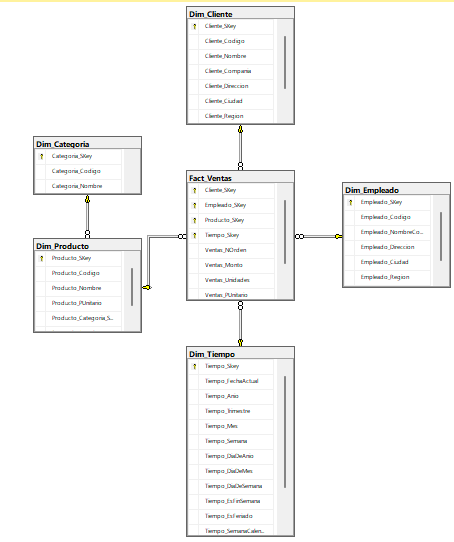
[Comando OLDB Insertar Registros Modificados 41](#_Toc142000574)

[Comando OLDB Actualizar Registros Modificados 42](#_Toc142000575)

# IMPLEMENTAR CARGA DIM EMPLEADO

Crear Base de Datos **DATAMART\_NORTHWND**.

## **MODELO COPO DE NIEVE**



## ORIGEN 1 STAGE DIM EMPLEADO

SELECT [Empleado\_Codigo]

,[Empleado\_NombreCompleto]

,[Empleado\_Direccion]

,[Empleado\_Ciudad]

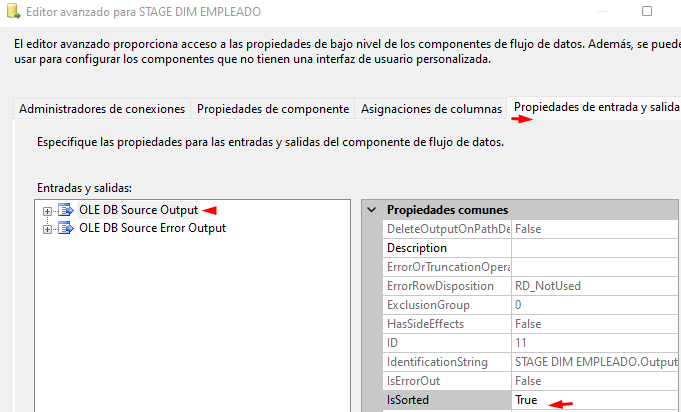
,[Empleado\_Region]

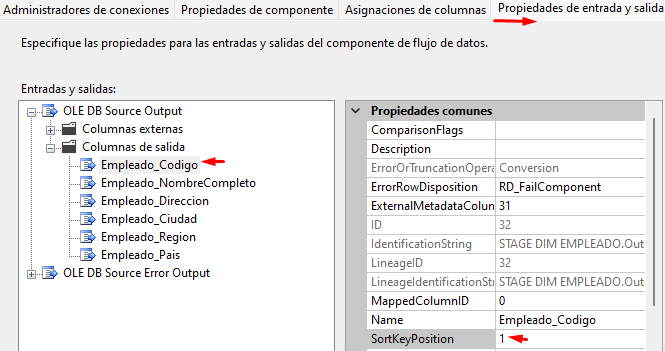
,[Empleado\_Pais]

FROM [STAGE\_NORTHWND].[dbo].[Stage\_Empleado]

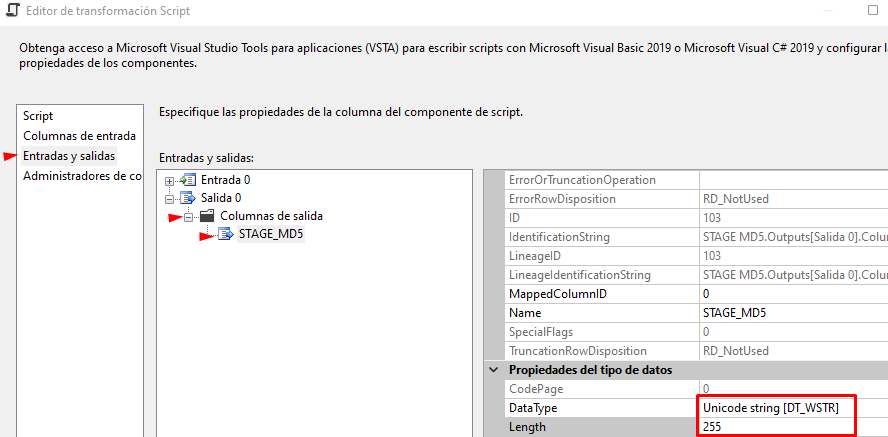
ORDER BY [Empleado\_Codigo]

### EDITOR AVANZADO





## COMPONENTE DE SCRIPT



Imports Microsoft.SqlServer.Dts.Pipeline

Imports System.Text

Imports System.Security.Cryptography

Private inputBuffer As PipelineBuffer

Public Overrides Sub ProcessInput(ByVal InputID As Integer, ByVal Buffer As Microsoft.SqlServer.Dts.Pipeline.PipelineBuffer)

inputBuffer = Buffer

MyBase.ProcessInput(InputID, Buffer)

End Sub

Public Shared Function CreateHash(ByVal data As String) As String

Dim dataToHash As Byte() = (New UnicodeEncoding()).GetBytes(data)

Dim md5 As MD5 = New MD5CryptoServiceProvider()

Dim hashedData As Byte() = md5.ComputeHash(dataToHash)

RNGCryptoServiceProvider.Create().GetBytes(dataToHash)

Dim s As String = Convert.ToBase64String(hashedData, Base64FormattingOptions.None)

Return s

End Function

Public Overrides Sub Entrada0\_ProcessInputRow(ByVal Row As Entrada0Buffer)

Dim counter As Integer = 0

Dim values As New StringBuilder

For counter = 0 To inputBuffer.ColumnCount - 1

Dim value As Object

value = inputBuffer.Item(counter)

values.Append(value)

Next

'CAMBIAR EL VALOR VariableSalida A SU COLUMNA DE SALIDA

Row.STAGEMD5 = CreateHash(values.ToString())

End Sub

## ORIGEN 2 DATAMART\_NORTHWND DIM EMPLEADO

SELECT [Empleado\_SKey]

,[Empleado\_Codigo]

,[Empleado\_NombreCompleto]

,[Empleado\_Direccion]

,[Empleado\_Ciudad]

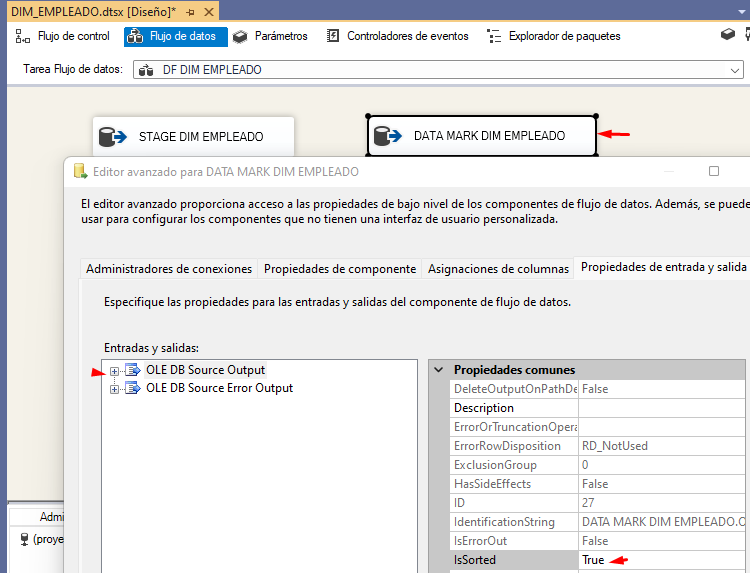
,[Empleado\_Region]

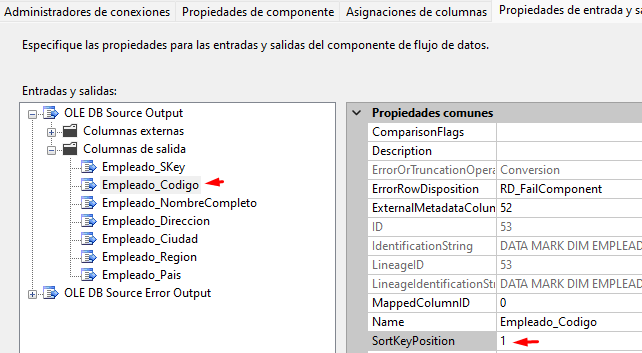
,[Empleado\_Pais]

FROM [DATAMART\_NORTHWND].[dbo].[Dim\_Empleado]

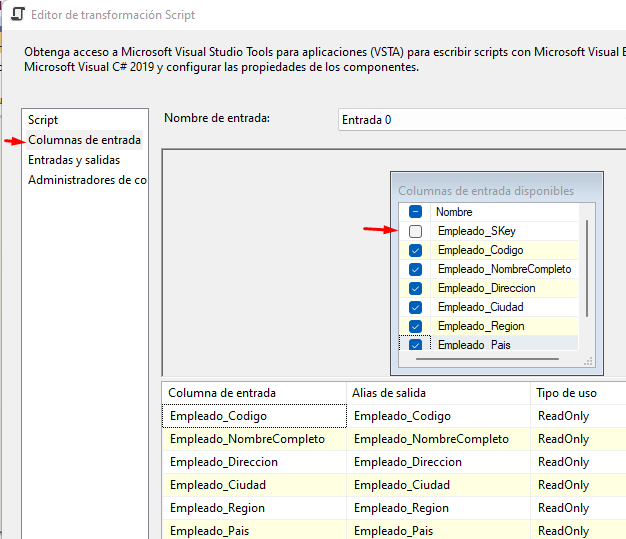
ORDER BY Empleado\_Codigo

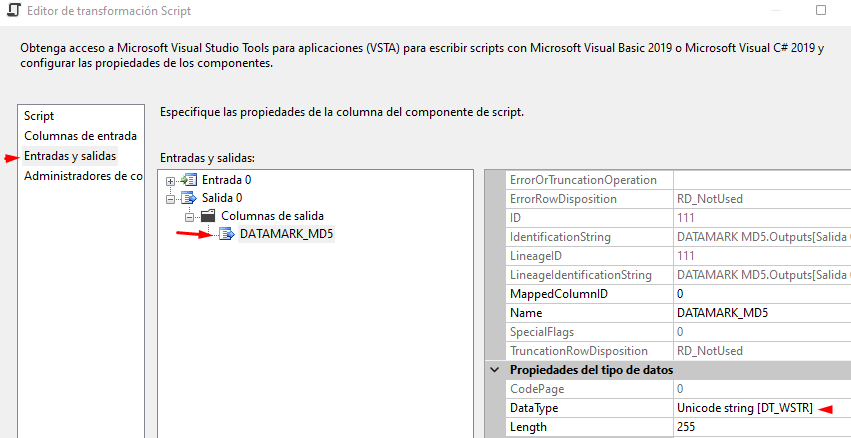
## EDITOR AVANZADO



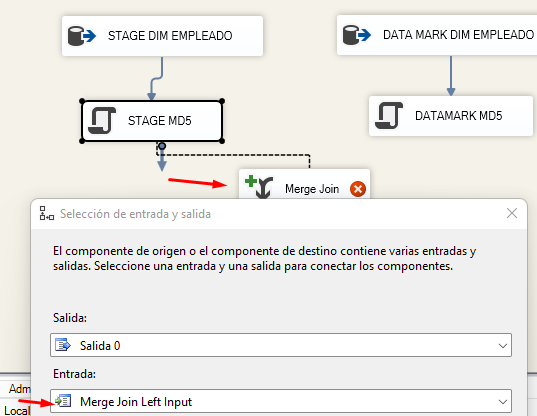


## COMPONENTE DE SCRIPT

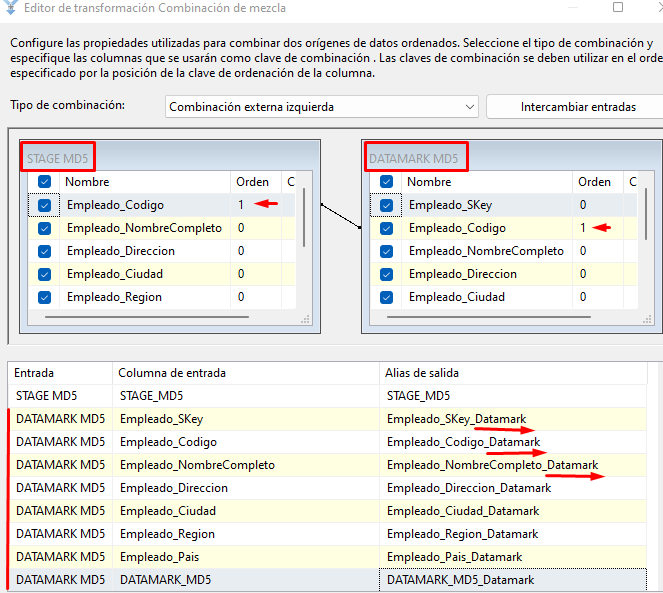




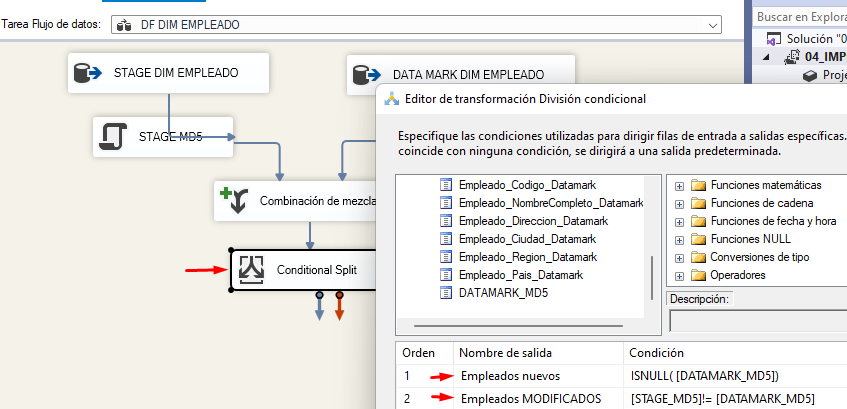
## COMBINACION DE MEZCLA



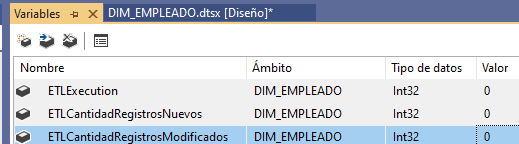
Simulación de left join

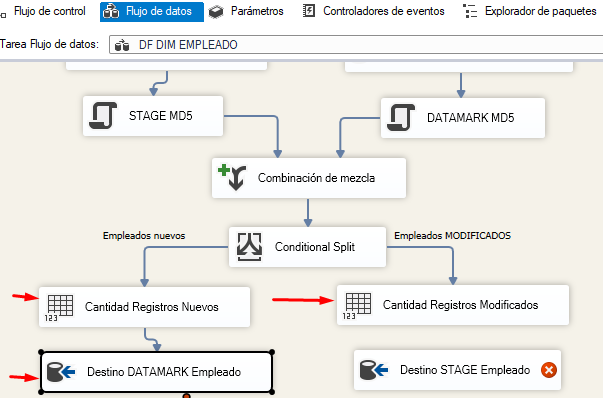


Especificar si son Empleados Nuevos o Modificados



Crear variables





Crear tabla en Base de datos **Stage\_Northwnd**

CREATE TABLE [STAGE\_NORTHWND].[dbo].[Dim\_Empleado\_Mod](

[IDEmpleado] [int] IDENTITY(1,1) NOT NULL,

[Empleado\_Codigo] [int] NOT NULL,

[Empleado\_NombreCompleto] [varchar](70) NOT NULL,

[Empleado\_Direccion] [varchar](120) NULL,

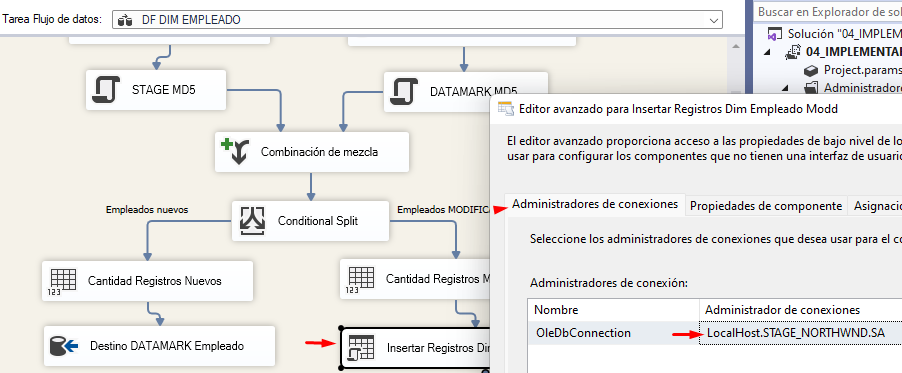
[Empleado\_Ciudad] [varchar](15) NULL,

[Empleado\_Region] [varchar](15) NULL,

[Empleado\_Pais] [varchar](15) NULL,

[ETLLoad] DATETIME )

Insertar registros en la tabla **Dim\_Empleado\_Mod.**



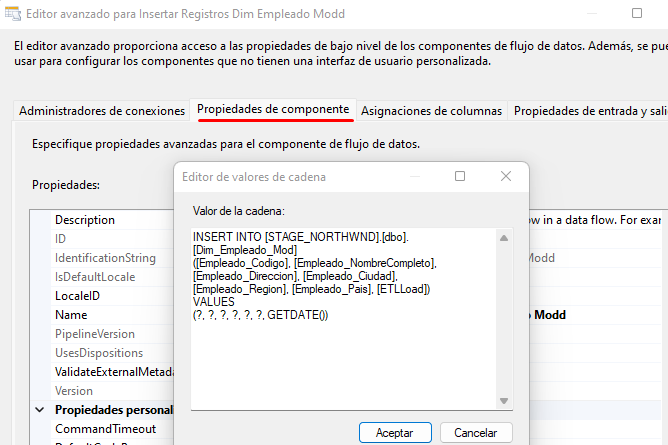
--Insertar registros en la tabla [STAGE\_NORTHWND].[dbo].[Dim\_Empleado\_Mod]

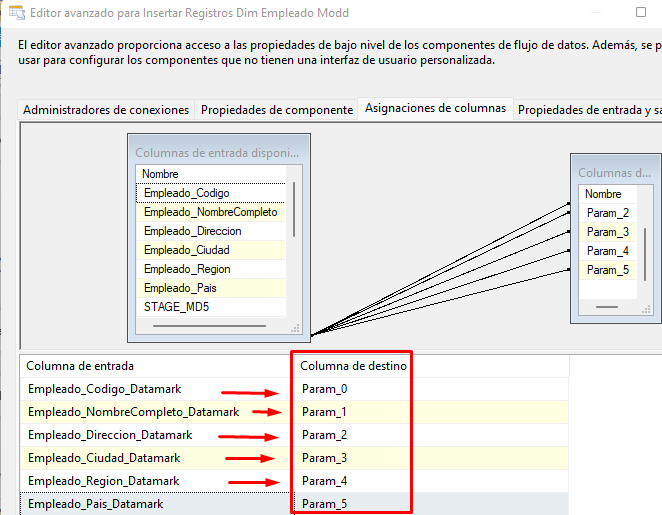
INSERT INTO [STAGE\_NORTHWND].[dbo].[Dim\_Empleado\_Mod]

([Empleado\_Codigo], [Empleado\_NombreCompleto], [Empleado\_Direccion], [Empleado\_Ciudad], [Empleado\_Region], [Empleado\_Pais], [ETLLoad])

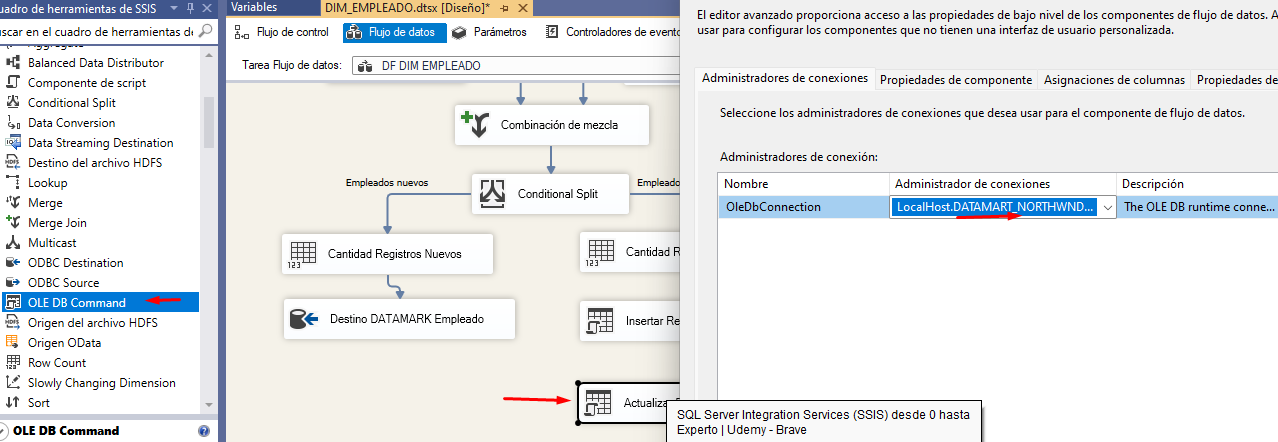
VALUES

(?, ?, ?, ?, ?, ?, GETDATE())





Actualizar Registros DATAMARK



UPDATE M

SET M.Empleado\_NombreCompleto = S.Empleado\_NombreCompleto,

M.Empleado\_Direccion = S.Empleado\_Direccion,

M.Empleado\_Ciudad = S.Empleado\_Ciudad,

M.Empleado\_Region = S.Empleado\_Region,

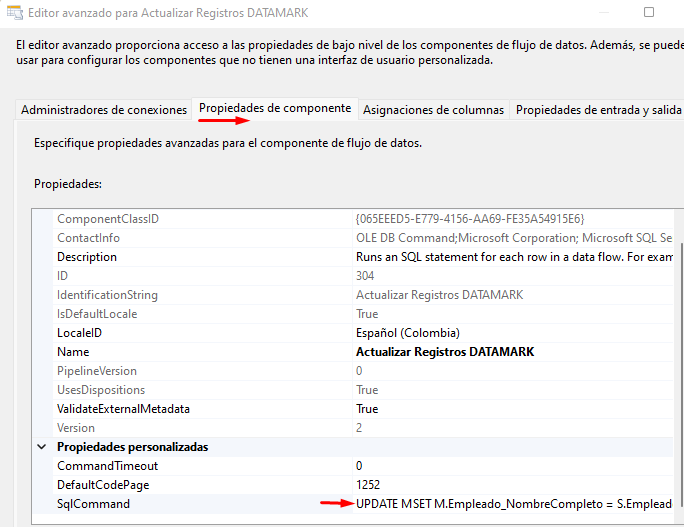
M.Empleado\_Pais = S.Empleado\_Pais

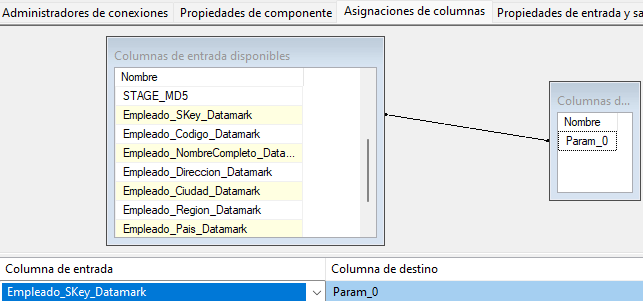
FROM [DATAMART\_NORTHWND].[dbo].[Dim\_Empleado] AS M

JOIN [STAGE\_NORTHWND].[dbo].[Stage\_Empleado] AS S

ON M.Empleado\_Codigo = S.Empleado\_Codigo

WHERE M.Empleado\_SKey = ?

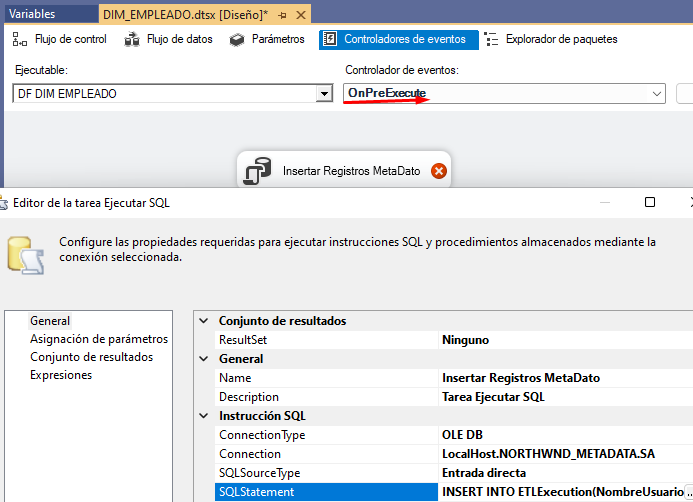




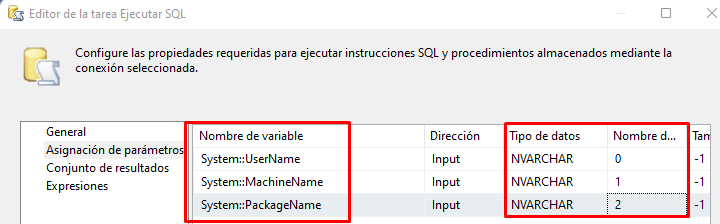
## Insertar en Metadatos Pre-Execute

INSERT INTO ETLExecution(NombreUsuario, NombreMaquina, NombrePaquete, ETLload)

VALUES (?,?,?,GETDATE())



Asignar Parámetros

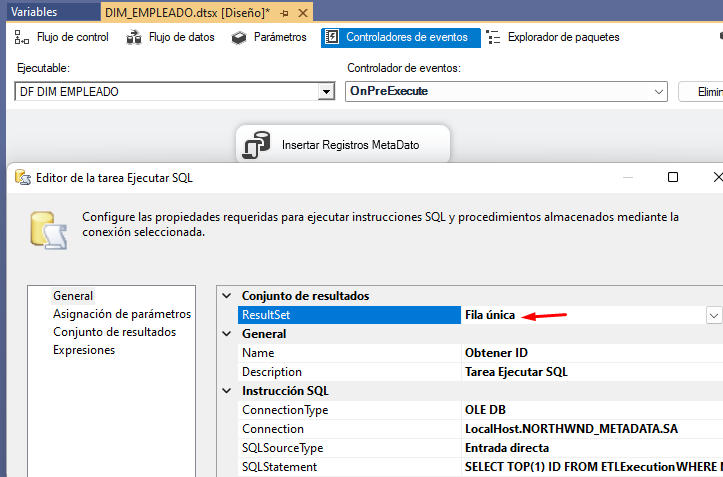


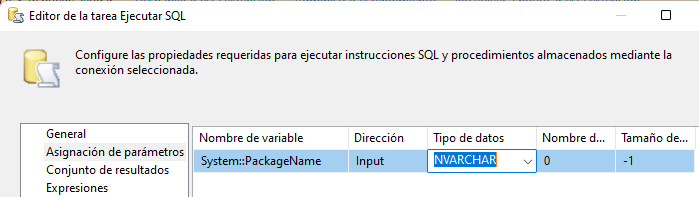
Obtener id de Metadata

SELECT TOP(1) ID FROM ETLExecution

WHERE NombrePaquete=?

Order By ID Desc







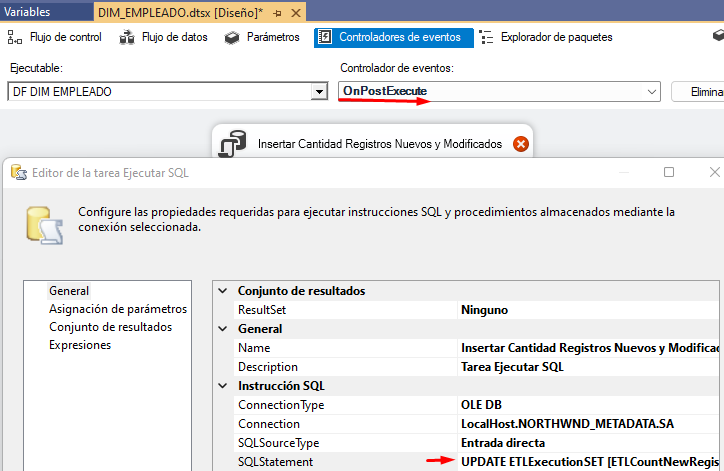
## Insertar en Metadatos Post-Execute

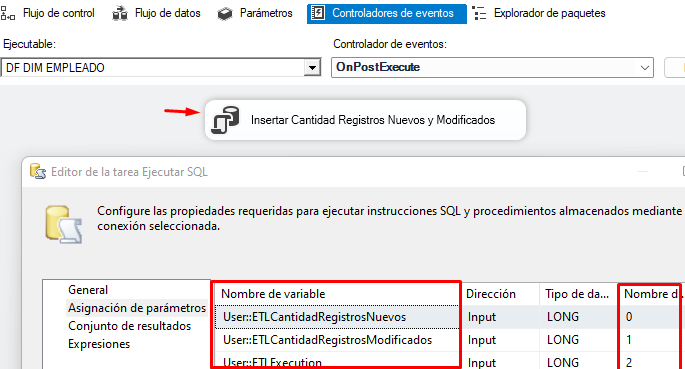
UPDATE ETLExecution

SET [ETLCountNewRegister] = ?,

[ETLCountModifiedRegister] = ?

WHERE ID = ?



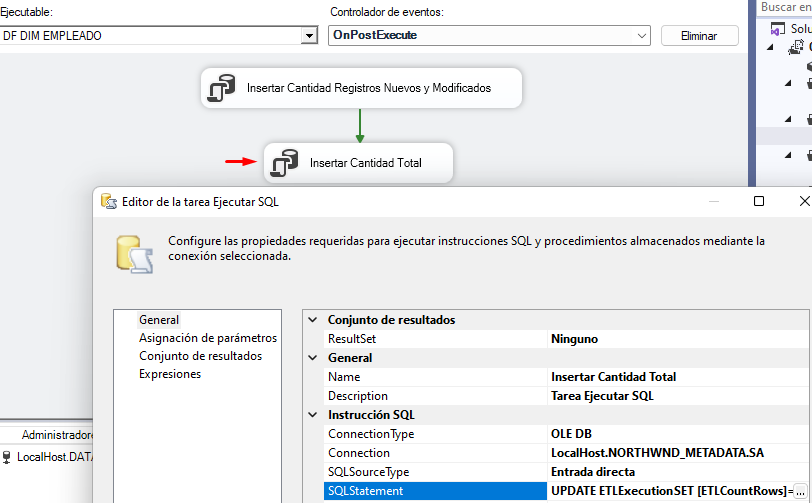


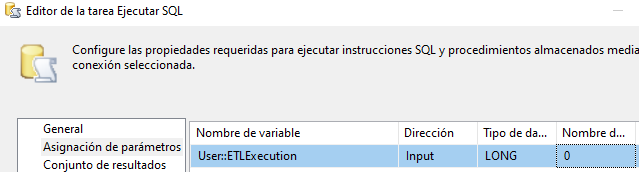
Insertar Valor total

UPDATE ETLExecution

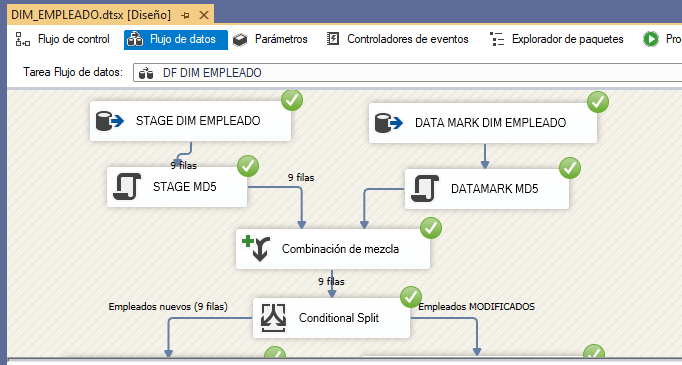
SET [ETLCountRows]=([ETLCountNewRegister]+[ETLCountModifiedRegister])

WHERE ID = ?

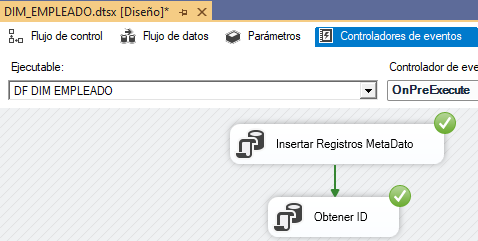




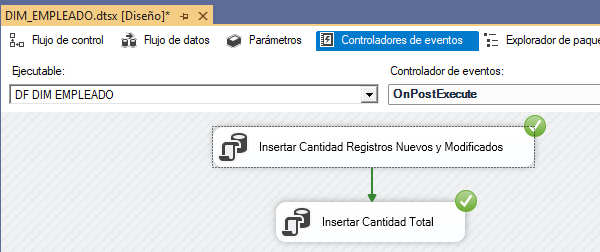
Prueba



Pre-Execute



Post-Execute



# IMPLEMENTAR CARGA DIM CATEGORIA

## Stage Categoria

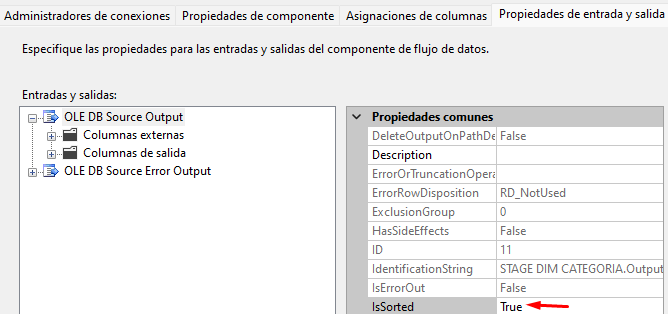
Se configura forma de ordenar.

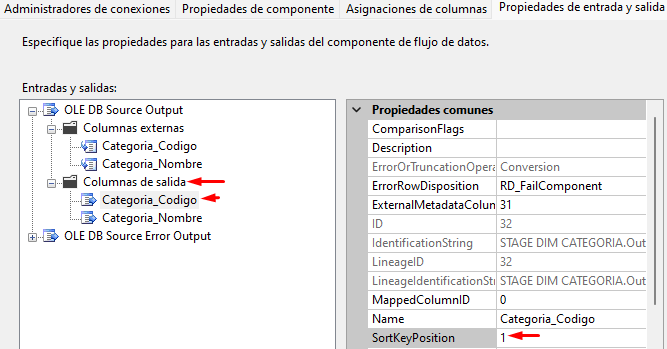
SELECT [Categoria\_Codigo],

[Categoria\_Nombre]

FROM [STAGE\_NORTHWND].[dbo].[Stage\_Categoria]

ORDER BY [Categoria\_Codigo]





## Data Mark Categoría

Se configura forma de ordenar igual que el anterior.

SELECT [Categoria\_SKey]

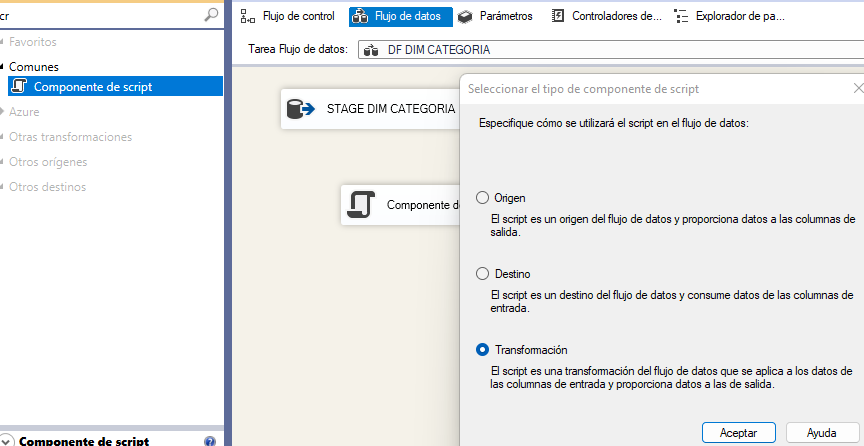
,[Categoria\_Codigo]

,[Categoria\_Nombre]

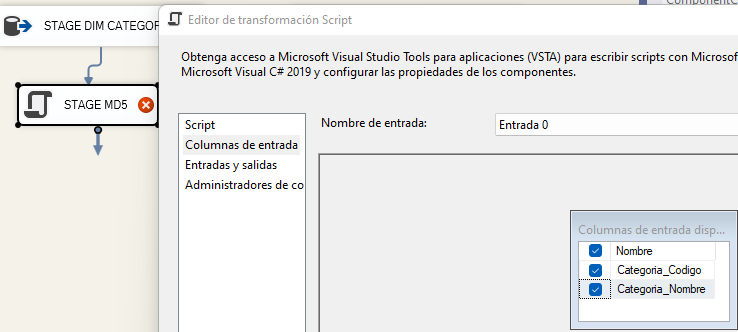
FROM [DATAMART\_NORTHWND].[dbo].[Dim\_Categoria]

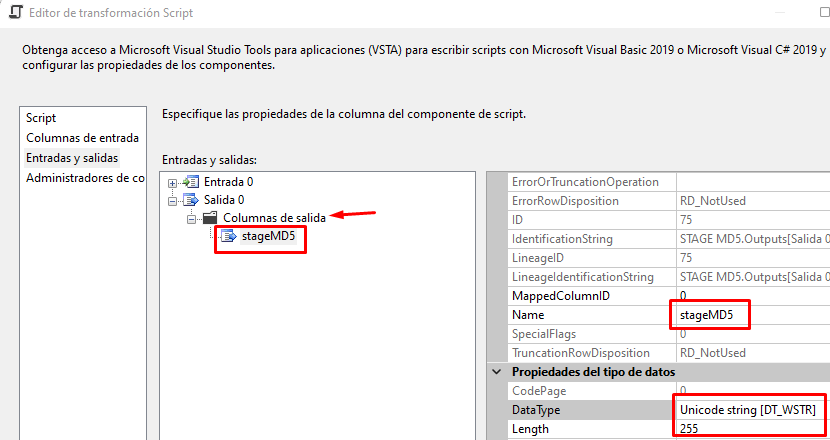
ORDER BY [Categoria\_Codigo]

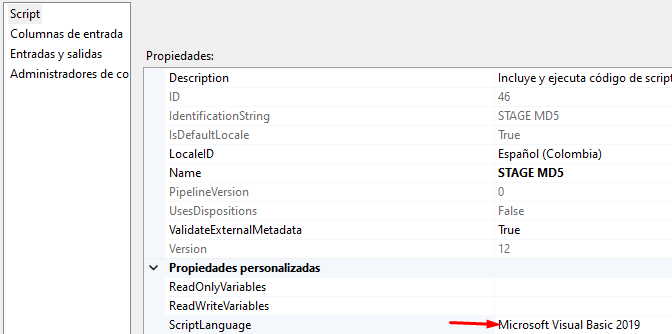
Hacer código de encriptación para **Stage y Data-Mark**.



### En transformación de Stage MD5







Imports Microsoft.SqlServer.Dts.Pipeline

Imports System.Text

Imports System.Security.Cryptography

**Antes de Metodo Entrada0\_ProcessInputRow pegar**

'''COPIAR Y PEGAR LOS METODOS

Private inputBuffer As PipelineBuffer

Public Overrides Sub ProcessInput(ByVal InputID As Integer, ByVal Buffer As Microsoft.SqlServer.Dts.Pipeline.PipelineBuffer)

inputBuffer = Buffer

MyBase.ProcessInput(InputID, Buffer)

End Sub

Public Shared Function CreateHash(ByVal data As String) As String

Dim dataToHash As Byte() = (New UnicodeEncoding()).GetBytes(data)

Dim md5 As MD5 = New MD5CryptoServiceProvider()

Dim hashedData As Byte() = md5.ComputeHash(dataToHash)

RNGCryptoServiceProvider.Create().GetBytes(dataToHash)

Dim s As String = Convert.ToBase64String(hashedData, Base64FormattingOptions.None)

Return s

End Function

**Dentro del** **Metodo Entrada0\_ProcessInputRow pegar**

Public Overrides Sub Entrada0\_ProcessInputRow(ByVal Row As Entrada0Buffer)

'''COPIAR Y PEGAR DENTRO DE LA ENTRADA DE PROCESOS

Dim counter As Integer = 0

Dim values As New StringBuilder

For counter = 0 To inputBuffer.ColumnCount - 1

Dim value As Object

value = inputBuffer.Item(counter)

values.Append(value)

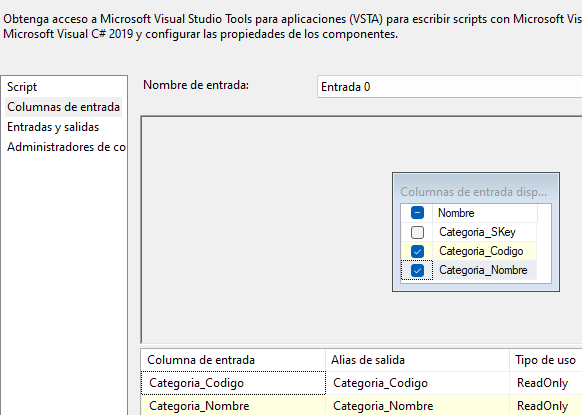
Next

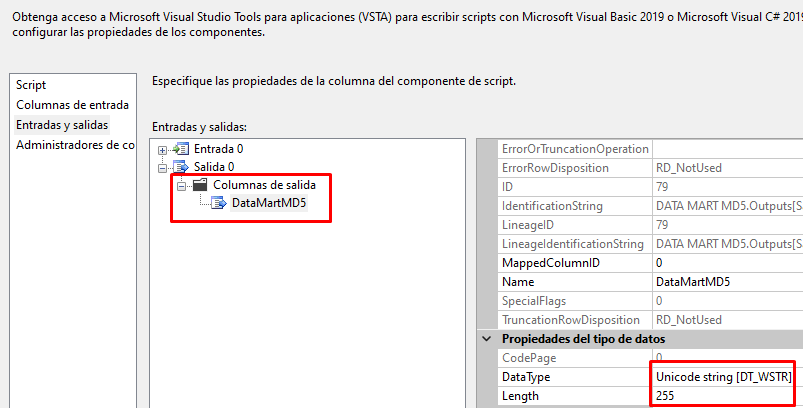
'CAMBIAR EL VALOR VariableSalida A SU COLUMNA DE SALIDA

Row.stageMD5 = CreateHash(values.ToString())

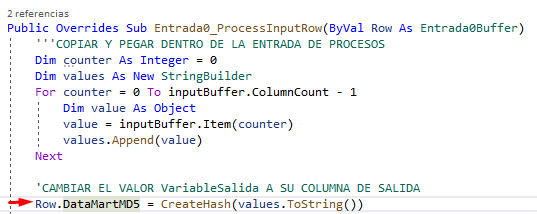
End Sub

### En transformación de Data-Mart MD5



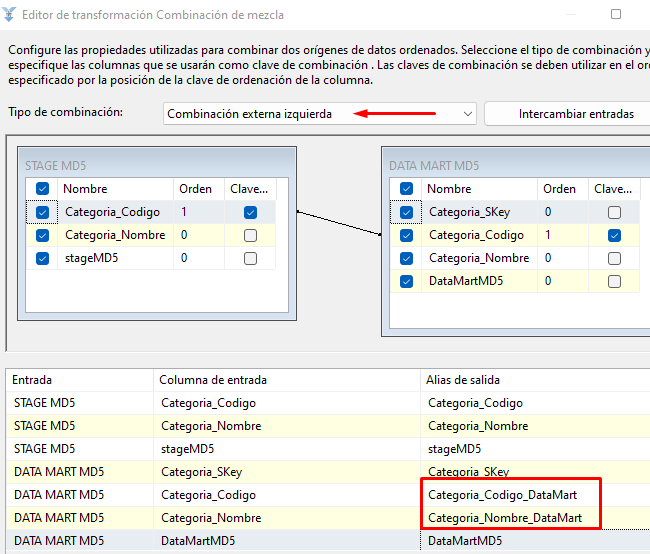


Hacer lo mismo en script del paso anterior.

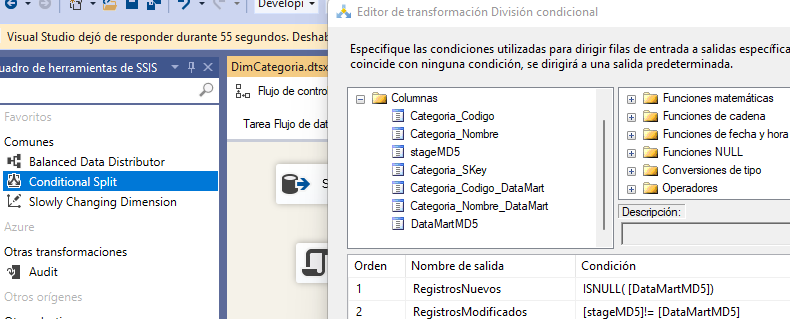


### Combinación de mezcla Stage izquierda y Data-Mart Derecha

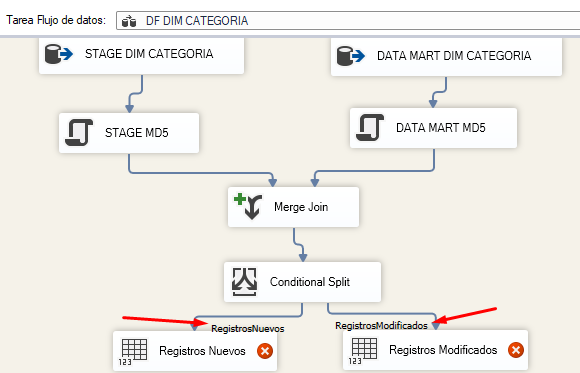
Es como si fuera un left join



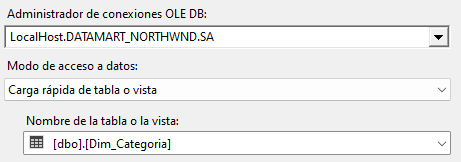
### División Condicional

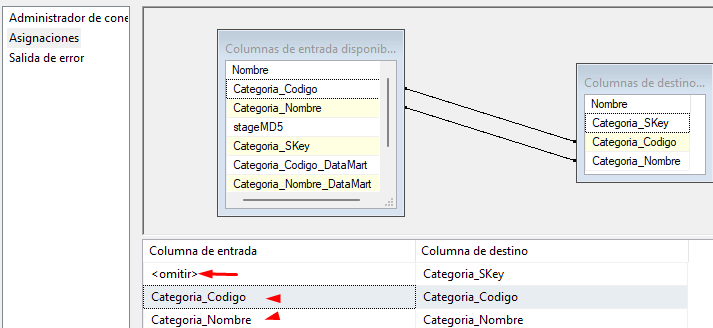


### Contar Filas

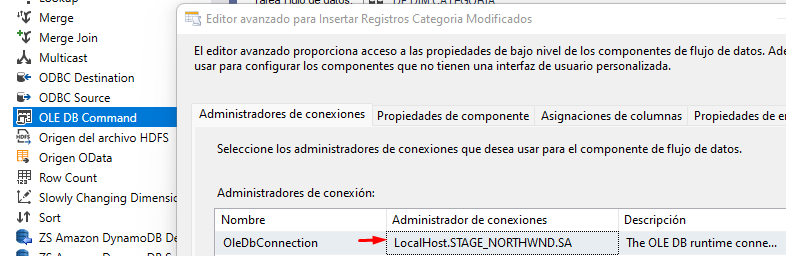


### Destino Data – Mart Categoría (nuevos)





### Comando OLDB Insertar Registros Modificados



--Crear la tabla [STAGE\_NORTHWND].[dbo].[Dim\_Categoria\_Mod]

CREATE TABLE [STAGE\_NORTHWND].[dbo].[Dim\_Categoria\_Mod](

[CategoriaID] [int] IDENTITY(1,1) NOT NULL,

[Categoria\_Codigo] [int] NOT NULL,

[Categoria\_Nombre] [varchar](15) NOT NULL,

[ETLLoad] [DATETIME]

)

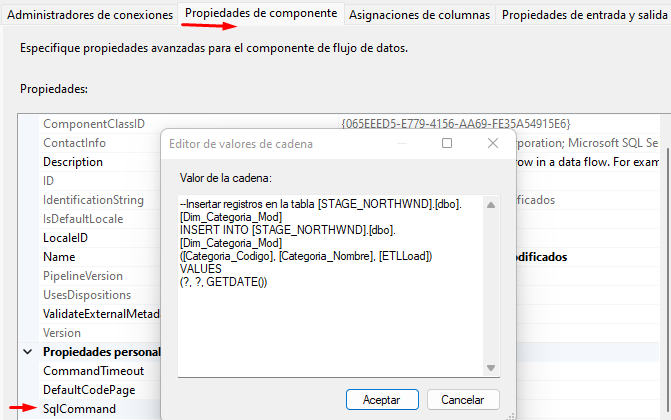
--Insertar registros en la tabla [STAGE\_NORTHWND].[dbo].[Dim\_Categoria\_Mod]

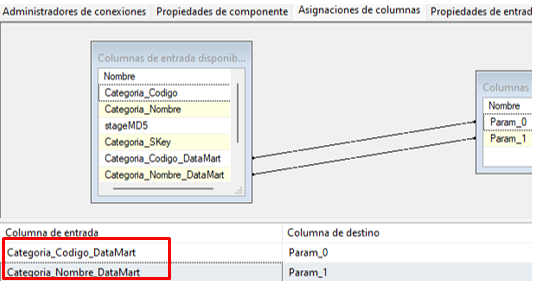
INSERT INTO [STAGE\_NORTHWND].[dbo].[Dim\_Categoria\_Mod]

([Categoria\_Codigo], [Categoria\_Nombre], [ETLLoad])

VALUES

(?, ?, GETDATE())





### Comando OLDB Actualizar Registros Modificados

--Actualizar la tabla [DATAMART\_NORTHWND].[dbo].[Dim\_Categoria]

--A partir de la tabla [STAGE\_NORTHWND].[dbo].[Stage\_Categoria]

UPDATE M

SET M.Categoria\_Nombre = S.Categoria\_Nombre

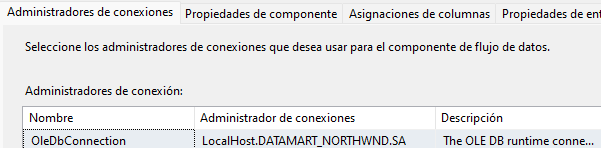
FROM [DATAMART\_NORTHWND].[dbo].[Dim\_Categoria] AS M

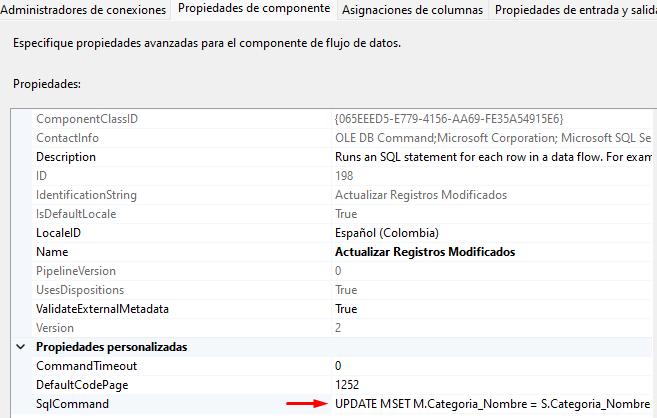
JOIN

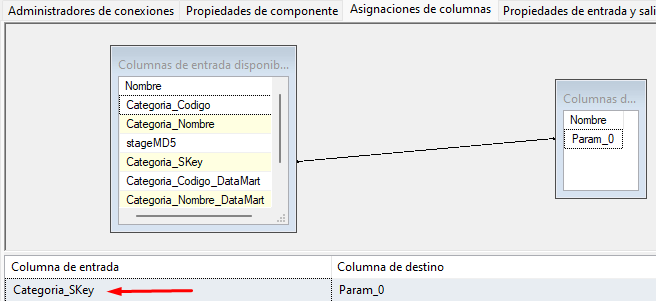
[STAGE\_NORTHWND].[dbo].[Stage\_Categoria] AS S

ON M.Categoria\_Codigo = S.Categoria\_Codigo

WHERE M.Categoria\_SKey = ?



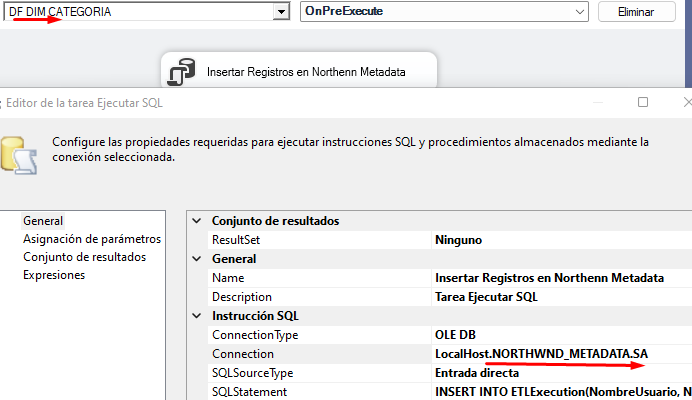




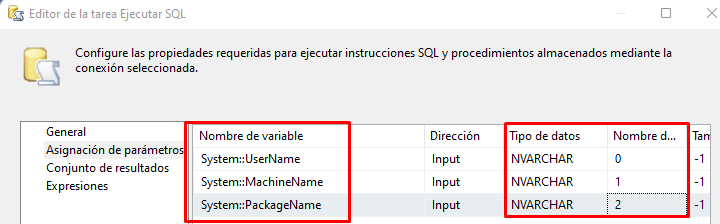
## Insertar en Metadatos Pre-Execute

INSERT INTO ETLExecution(NombreUsuario, NombreMaquina, NombrePaquete, ETLload)

VALUES (?,?,?,GETDATE())



Asignar Parámetros

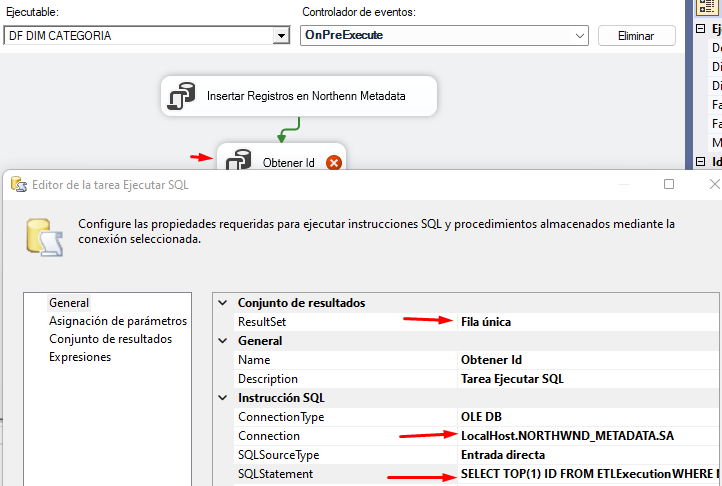


Obtener id de Metadata

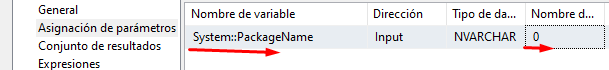
SELECT TOP(1) ID FROM ETLExecution

WHERE NombrePaquete=?

Order By ID Desc



Asignar Parametro





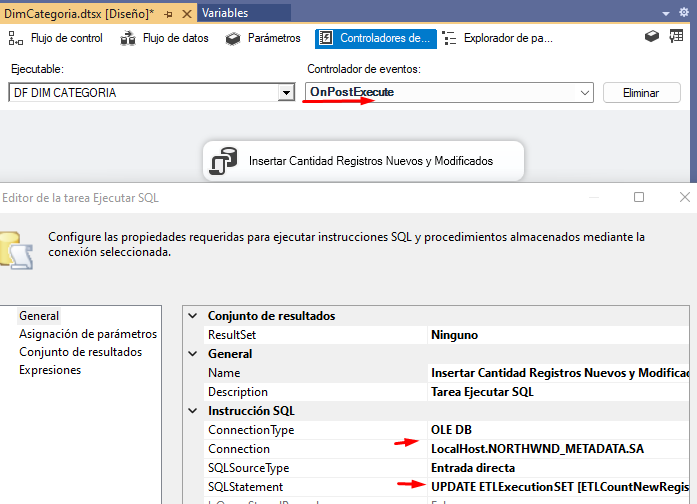
## Insertar en Metadatos Post-Execute

UPDATE ETLExecution

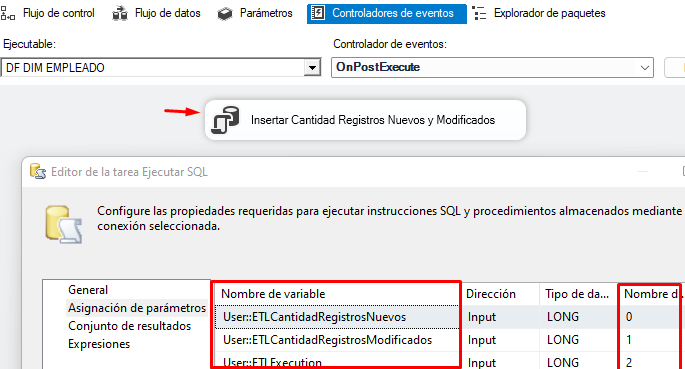
SET [ETLCountNewRegister] = ?,

[ETLCountModifiedRegister] = ?

WHERE ID = ?



Asignar Parametros

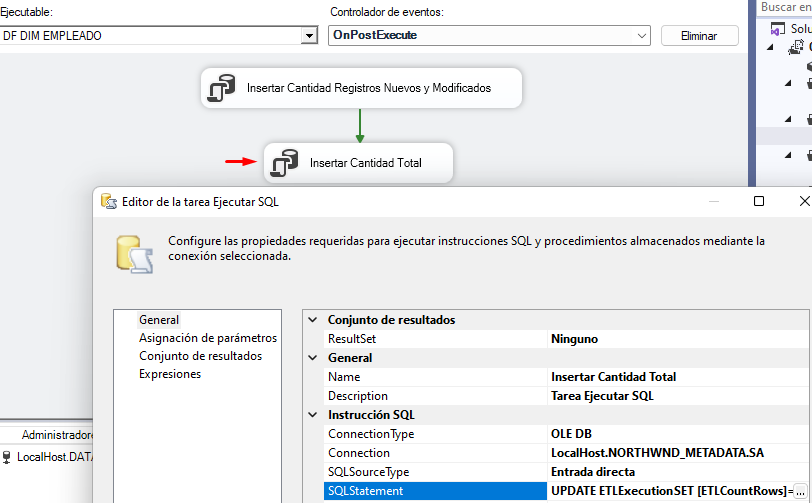


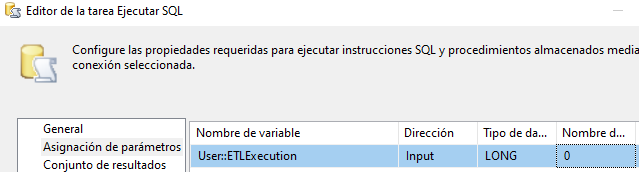
Insertar Valor total

UPDATE ETLExecution

SET [ETLCountRows]=([ETLCountNewRegister]+[ETLCountModifiedRegister])

WHERE ID = ?



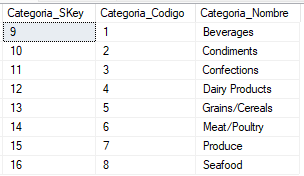


Prueba

SELECT \*

FROM [DATAMART\_NORTHWND].[dbo].[Dim\_Categoria]

ORDER BY [Categoria\_Codigo]



select \*

from [STAGE\_NORTHWND].[dbo].[Dim\_Categoria\_Mod]

SELECT \*

FROM [NORTHWND\_METADATA].[dbo].ETLExecution

WHERE NombrePaquete = 'Dim\_Categoria'



# IMPLEMENTAR CARGA DIM PRODUCTO

## Stage Producto

Se configura forma de ordenar.

SELECT [Producto\_Codigo]

,[Producto\_Nombre]

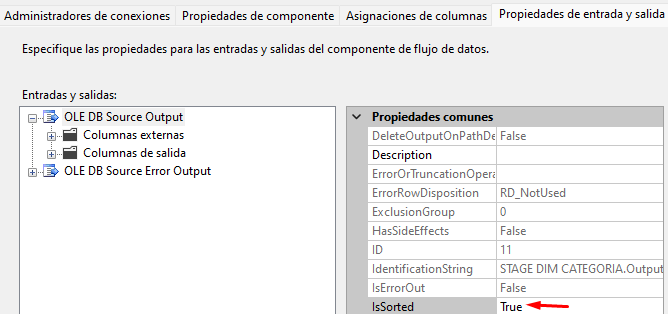
,[Producto\_PUnitario]

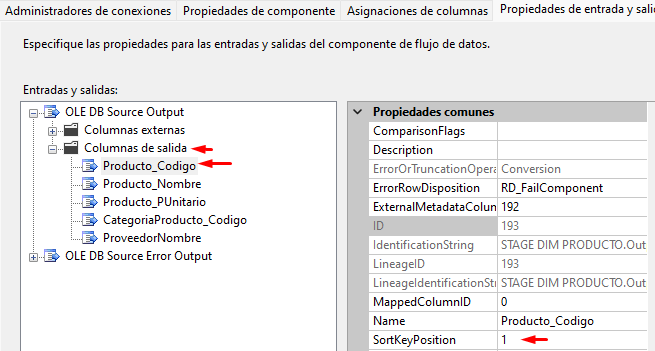
,[CategoriaProducto\_Codigo]

,[ProveedorNombre]

FROM [STAGE\_NORTHWND].[dbo].[Stage\_Producto]

ORDER BY [Producto\_Codigo]





## Data Mark Producto

Se configura forma de ordenar igual que el anterior.

SELECT [Producto\_SKey]

,[Producto\_Codigo]

,[Producto\_Nombre]

,[Producto\_PUnitario]

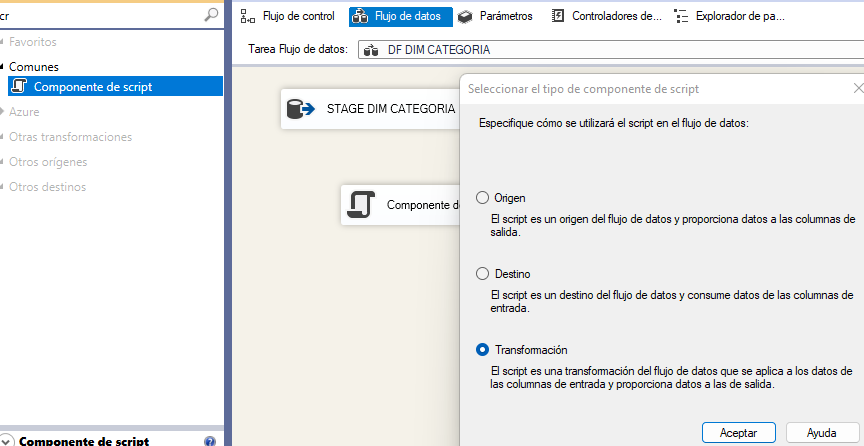
,[Producto\_Categoria\_SKey]

,[ProveedorNombre]

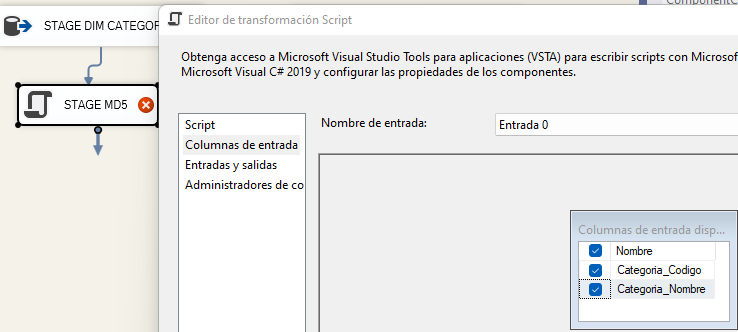
FROM [DATAMART\_NORTHWND].[dbo].[Dim\_Producto]

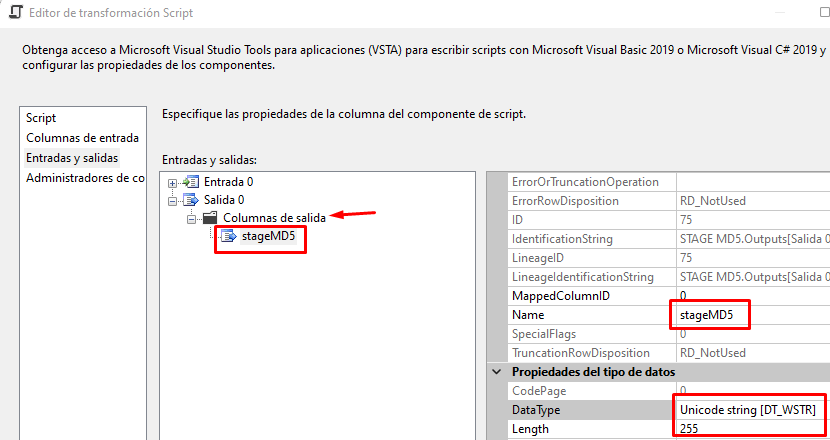
ORDER BY [Producto\_Codigo]

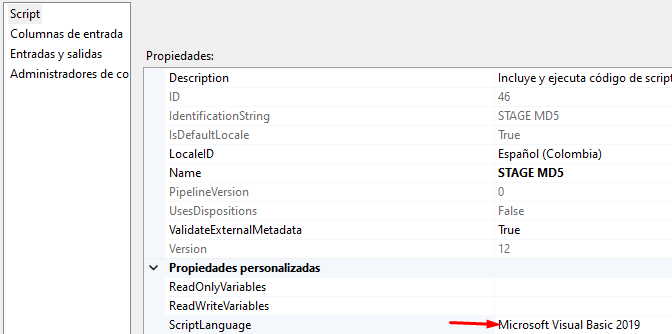
Hacer código de encriptación para **Stage y Data-Mark**.



### En transformación de Stage MD5







Imports Microsoft.SqlServer.Dts.Pipeline

Imports System.Text

Imports System.Security.Cryptography

**Antes de Metodo Entrada0\_ProcessInputRow pegar**

'''COPIAR Y PEGAR LOS METODOS

Private inputBuffer As PipelineBuffer

Public Overrides Sub ProcessInput(ByVal InputID As Integer, ByVal Buffer As Microsoft.SqlServer.Dts.Pipeline.PipelineBuffer)

inputBuffer = Buffer

MyBase.ProcessInput(InputID, Buffer)

End Sub

Public Shared Function CreateHash(ByVal data As String) As String

Dim dataToHash As Byte() = (New UnicodeEncoding()).GetBytes(data)

Dim md5 As MD5 = New MD5CryptoServiceProvider()

Dim hashedData As Byte() = md5.ComputeHash(dataToHash)

RNGCryptoServiceProvider.Create().GetBytes(dataToHash)

Dim s As String = Convert.ToBase64String(hashedData, Base64FormattingOptions.None)

Return s

End Function

**Dentro del** **Metodo Entrada0\_ProcessInputRow pegar**

Public Overrides Sub Entrada0\_ProcessInputRow(ByVal Row As Entrada0Buffer)

'''COPIAR Y PEGAR DENTRO DE LA ENTRADA DE PROCESOS

Dim counter As Integer = 0

Dim values As New StringBuilder

For counter = 0 To inputBuffer.ColumnCount - 1

Dim value As Object

value = inputBuffer.Item(counter)

values.Append(value)

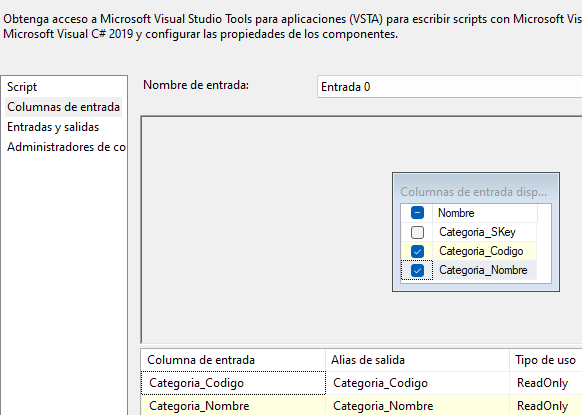
Next

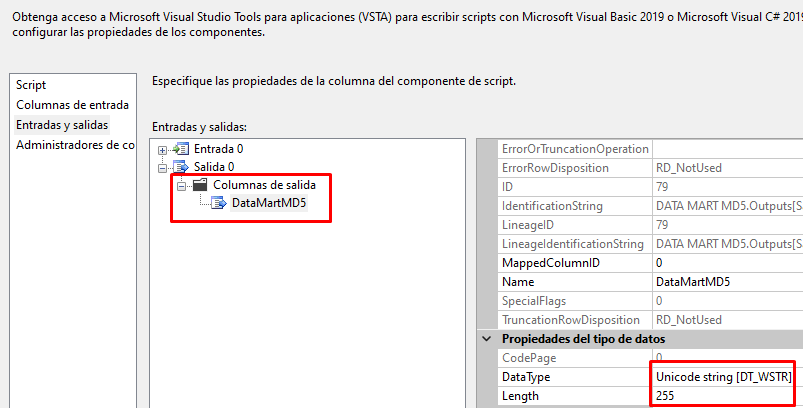
'CAMBIAR EL VALOR VariableSalida A SU COLUMNA DE SALIDA

Row.stageMD5 = CreateHash(values.ToString())

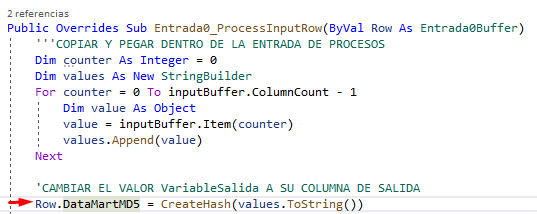
End Sub

### En transformación de Data-Mart MD5



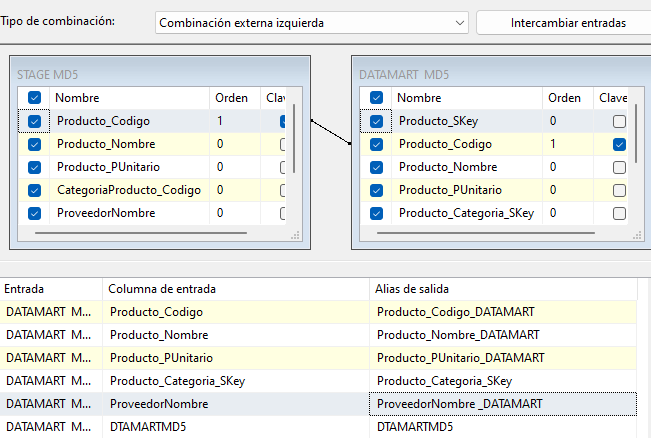


Hacer lo mismo en script del paso anterior.

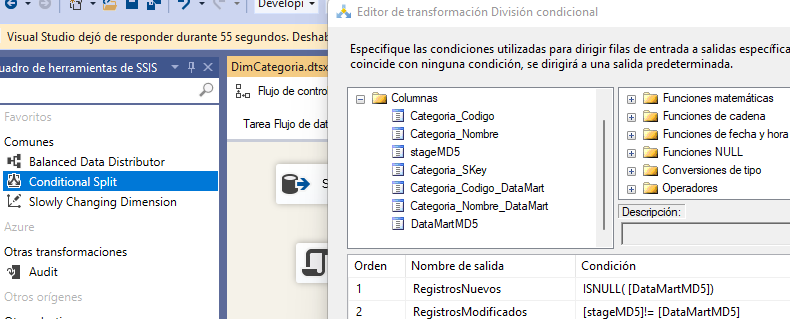


### Combinación de mezcla Stage izquierda y Data-Mart Derecha

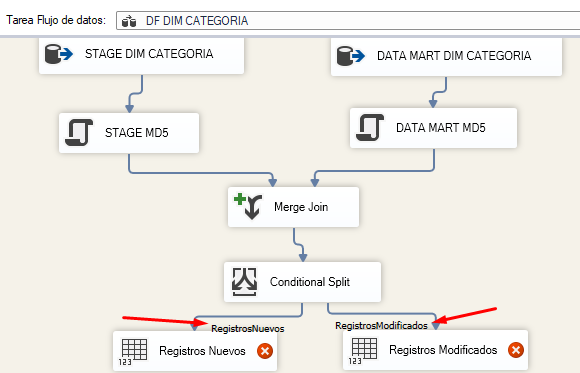
Es como si fuera un left join



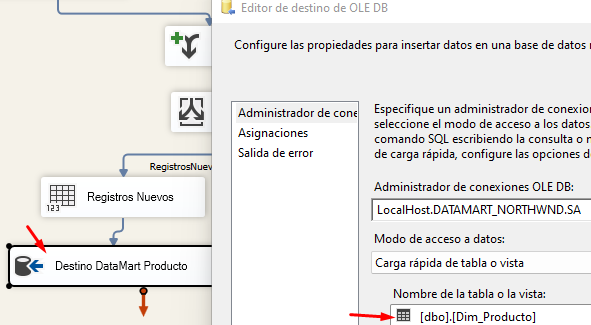
### División Condicional

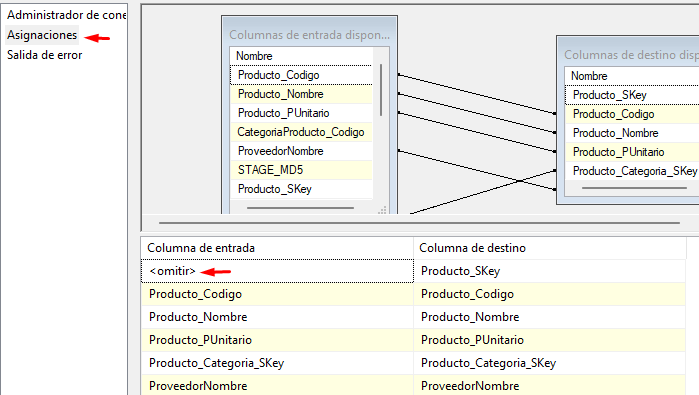


### Contar Filas

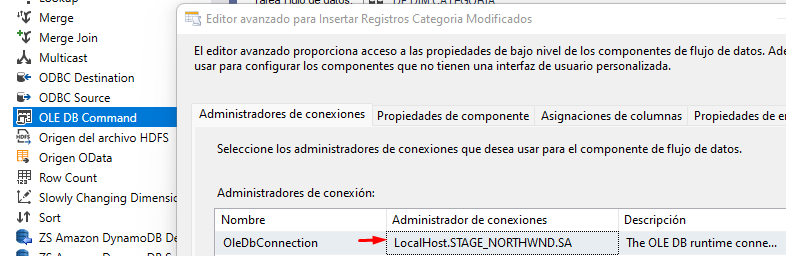


### Destino Data – Mart Categoría (nuevos)





### Comando OLDB Insertar Registros Modificados



CREATE TABLE [STAGE\_NORTHWND].[dbo].[Dim\_Producto\_Mod](

[ProductoID] [int] IDENTITY(1,1) NOT NULL,

[Producto\_Codigo] [int] NOT NULL,

[Producto\_Nombre] [varchar](80) NOT NULL,

[Producto\_PUnitario] [decimal](15, 2) NULL,

[Producto\_Categoria\_SKey] [int] NOT NULL,

[ProveedorNombre] [varchar](40) NULL,

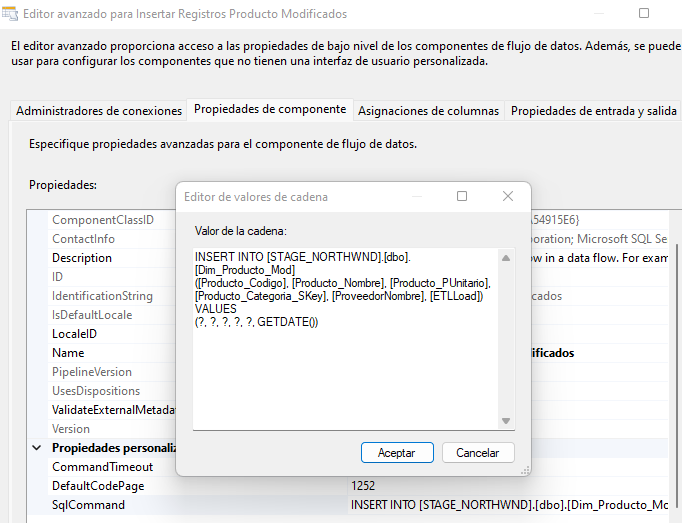
[ETLLoad] [DATETIME] )

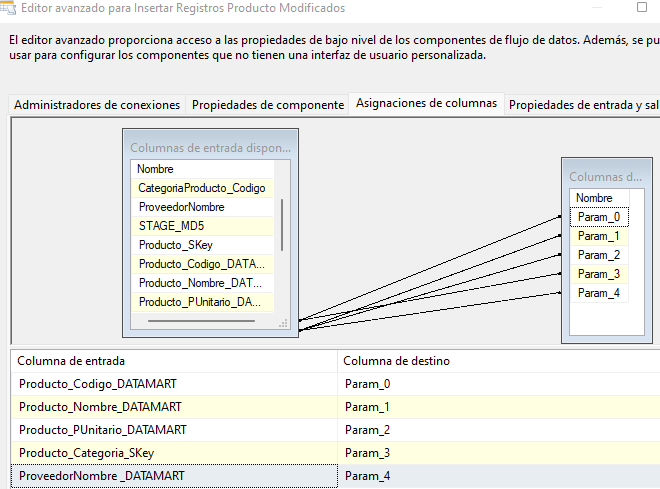
INSERT INTO [STAGE\_NORTHWND].[dbo].[Dim\_Producto\_Mod]

([Producto\_Codigo], [Producto\_Nombre], [Producto\_PUnitario], [Producto\_Categoria\_SKey], [ProveedorNombre], [ETLLoad])

VALUES

(?, ?, ?, ?, ?, GETDATE())





### Comando OLDB Actualizar Registros Modificados

UPDATE M

SET M.Producto\_Nombre = S.Producto\_Nombre,

M.Producto\_PUnitario = S.Producto\_PUnitario,

M.Producto\_Categoria\_SKey = S.CategoriaProducto\_Codigo,

M.ProveedorNombre = S.ProveedorNombre

FROM [DATAMART\_NORTHWND].[dbo].[Dim\_Producto] AS M

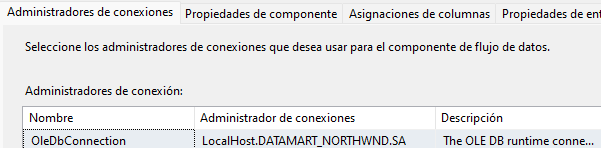
JOIN [STAGE\_NORTHWND].[dbo].[Stage\_Producto] AS S

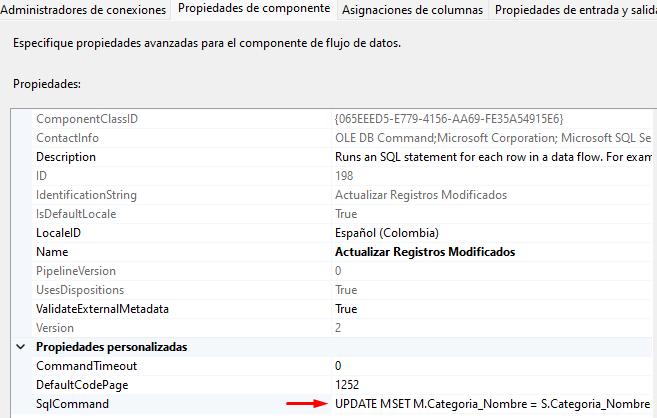
ON M.Producto\_Codigo = S.Producto\_Codigo

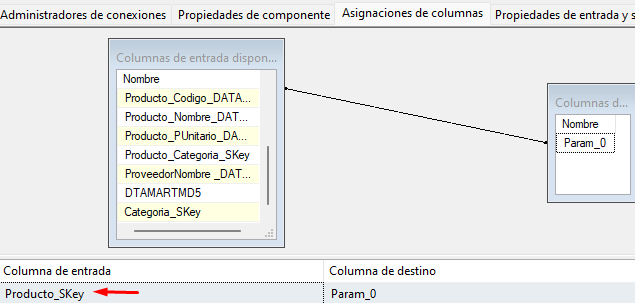
JOIN [DATAMART\_NORTHWND].[dbo].[Dim\_Categoria] AS C

ON S.CategoriaProducto\_Codigo = C.Categoria\_Codigo

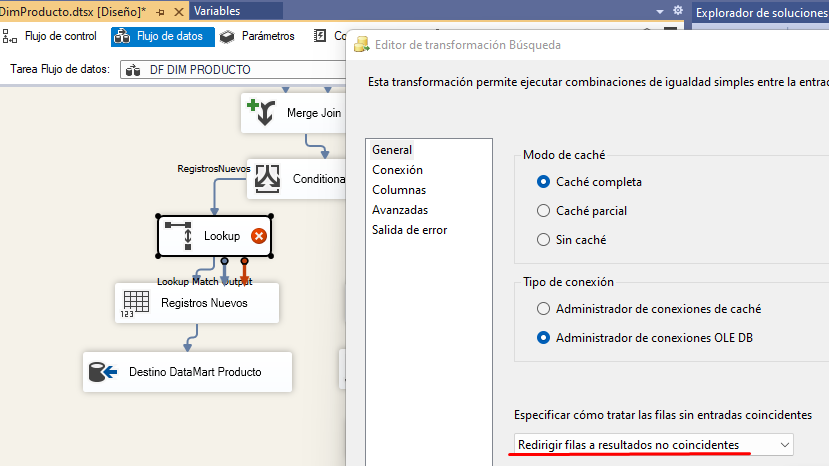
WHERE M.Producto\_SKey = ?







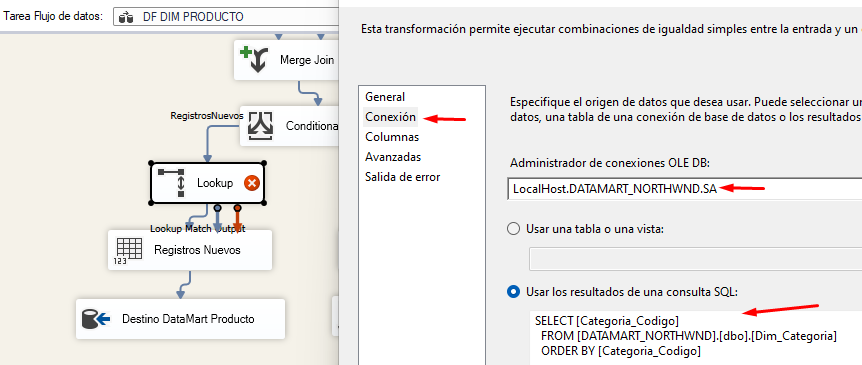
### Agregar Búsqueda Nuevo

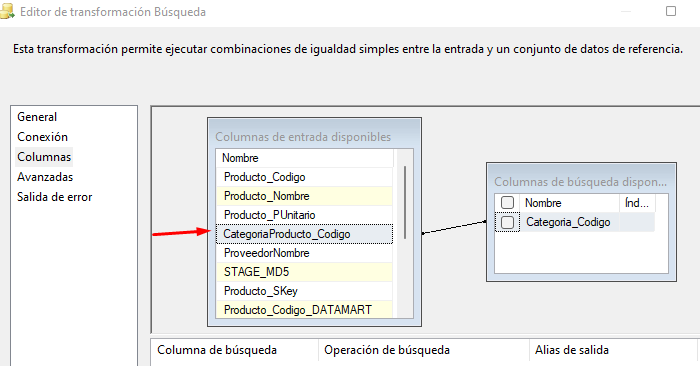


SELECT [Categoria\_Codigo]

FROM [DATAMART\_NORTHWND].[dbo].[Dim\_Categoria]

ORDER BY [Categoria\_Codigo]

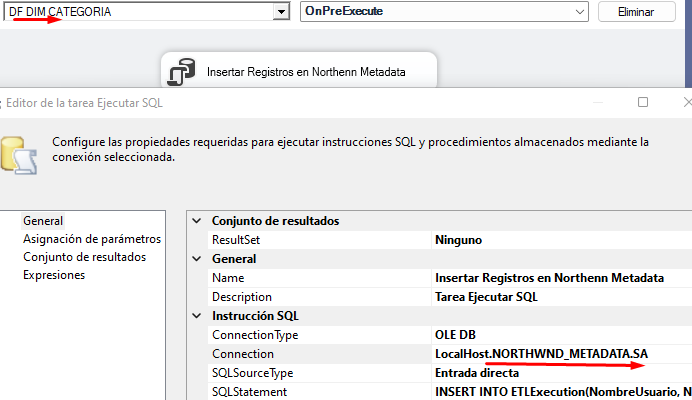




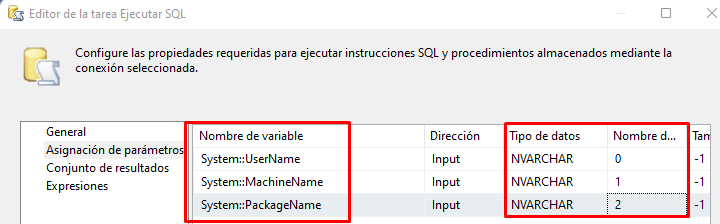
## Insertar en Metadatos Pre-Execute

INSERT INTO ETLExecution(NombreUsuario, NombreMaquina, NombrePaquete, ETLload)

VALUES (?,?,?,GETDATE())



Asignar Parámetros

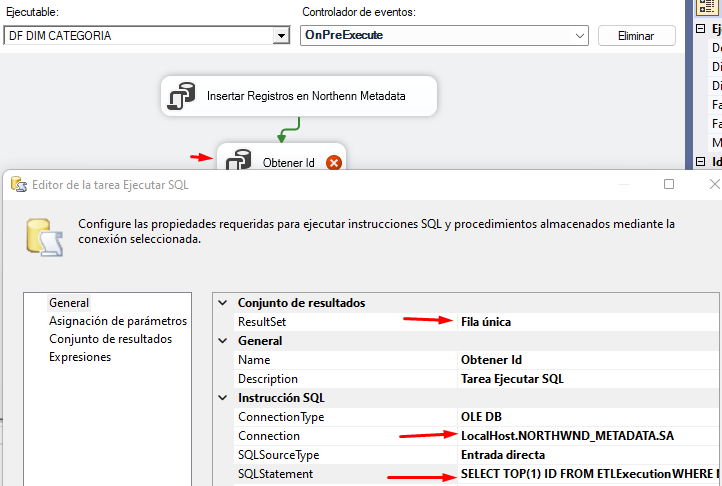


Obtener id de Metadata

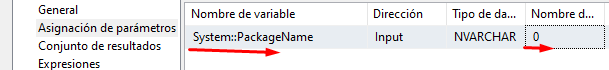
SELECT TOP(1) ID FROM ETLExecution

WHERE NombrePaquete=?

Order By ID Desc



Asignar Parametro





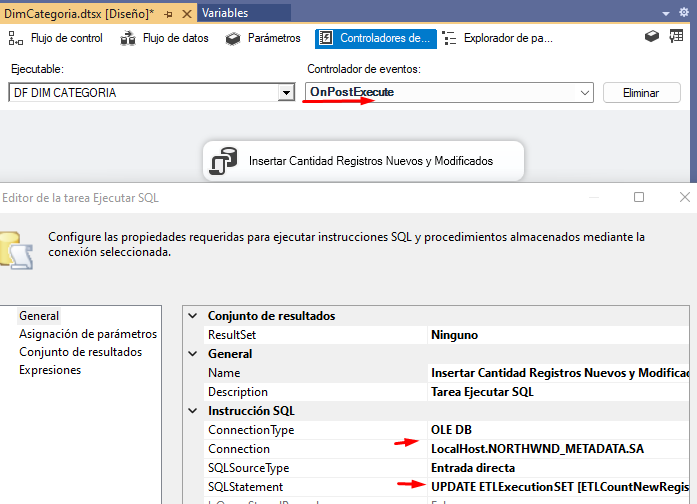
## Insertar en Metadatos Post-Execute

UPDATE ETLExecution

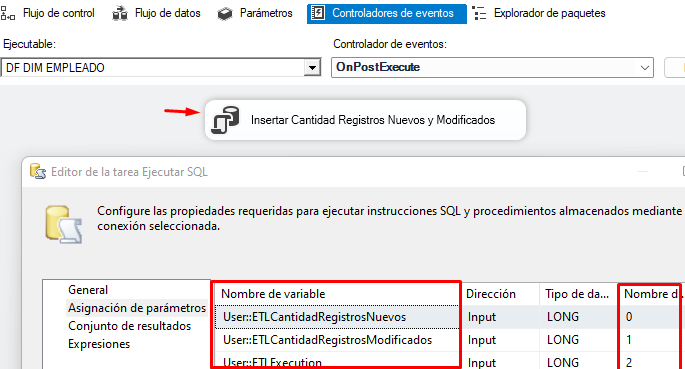
SET [ETLCountNewRegister] = ?,

[ETLCountModifiedRegister] = ?

WHERE ID = ?



Asignar Parametros

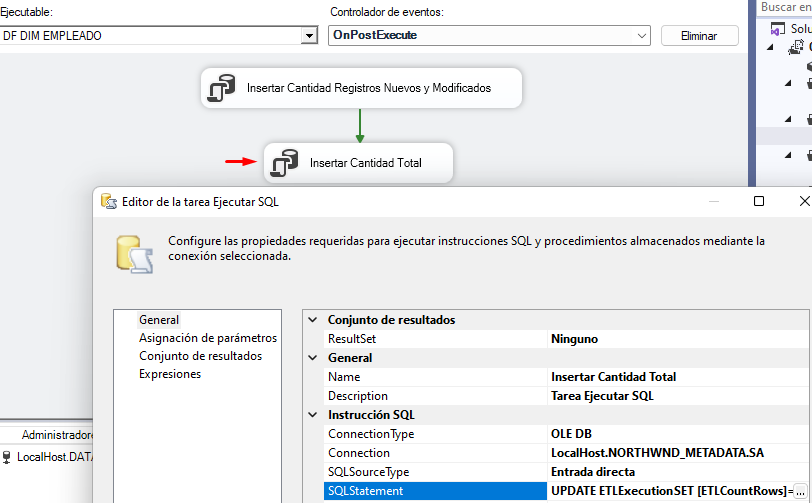


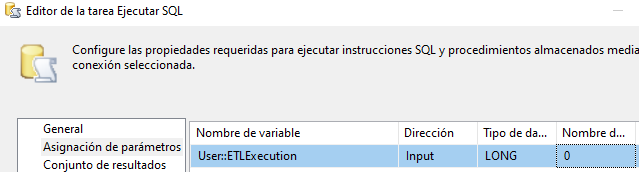
Insertar Valor total

UPDATE ETLExecution

SET [ETLCountRows]=([ETLCountNewRegister]+[ETLCountModifiedRegister])

WHERE ID = ?





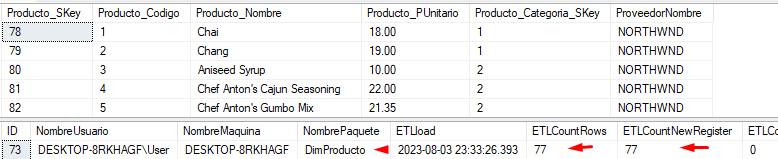
Prueba

SELECT \* FROM [DATAMART\_NORTHWND].[dbo].[Dim\_Producto]

SELECT \* FROM [NORTHWND\_METADATA].[dbo].[ETLExecution]

ORDER BY ID desc

SELECT \* FROM [STAGE\_NORTHWND].[dbo].[Dim\_Producto\_Mod]



# IMPLEMENTAR CARGA DIM CLIENTE

## Stage Cliente

Se configura forma de ordenar.

SELECT [Cliente\_Codigo]

,[Cliente\_Nombre]

,[Cliente\_Compania]

,[Cliente\_Direccion]

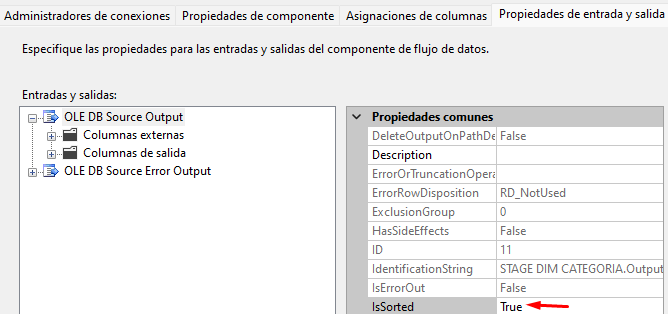
,[Cliente\_Ciudad]

,[Cliente\_Region]

,[Cliente\_Pais]

FROM [STAGE\_NORTHWND].[dbo].[Stage\_Cliente]

ORDER BY [Cliente\_Codigo]



## Data Mark Cliente

Se configura forma de ordenar igual que el anterior.

SELECT [Cliente\_SKey]

,[Cliente\_Codigo]

,[Cliente\_Nombre]

,[Cliente\_Compania]

,[Cliente\_Direccion]

,[Cliente\_Ciudad]

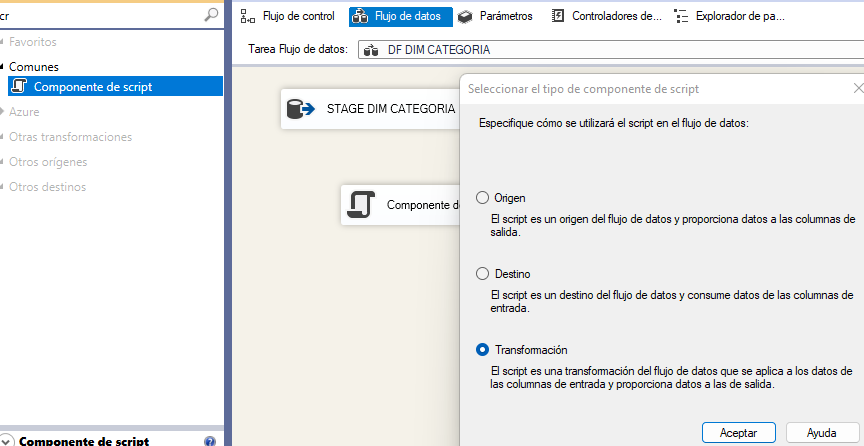
,[Cliente\_Region]

,[Cliente\_Pais]

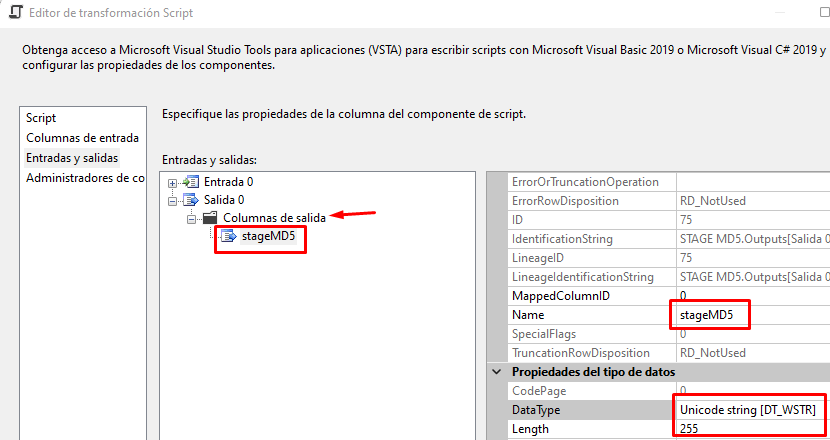
FROM [DATAMART\_NORTHWND].[dbo].[Dim\_Cliente]

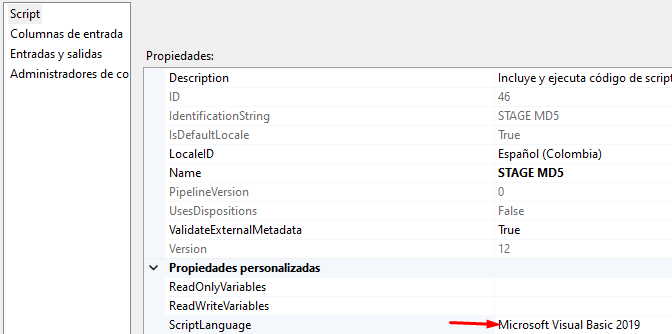
ORDER BY [Cliente\_Codigo]

Hacer código de encriptación para **Stage y Data-Mark**.



### En transformación de Stage MD5

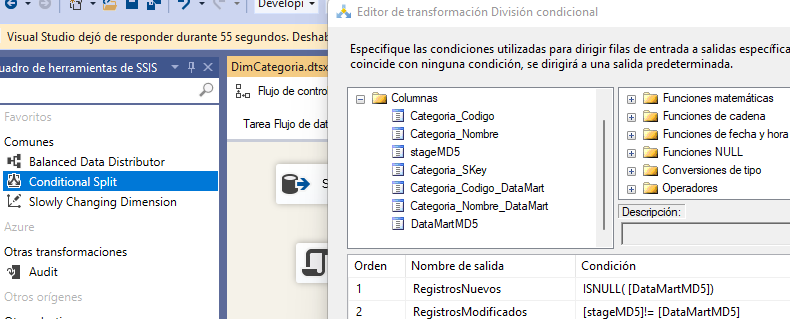




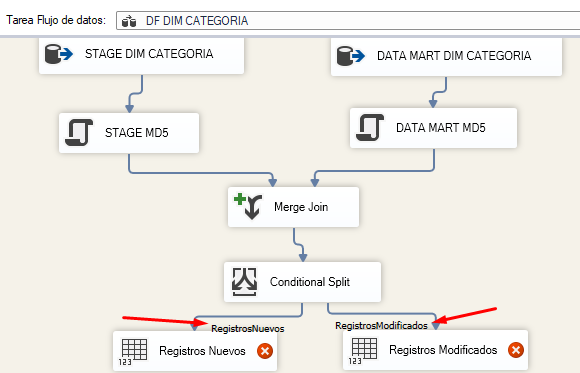
### Combinación de mezcla Stage izquierda y Data-Mart Derecha

Es como si fuera un left join

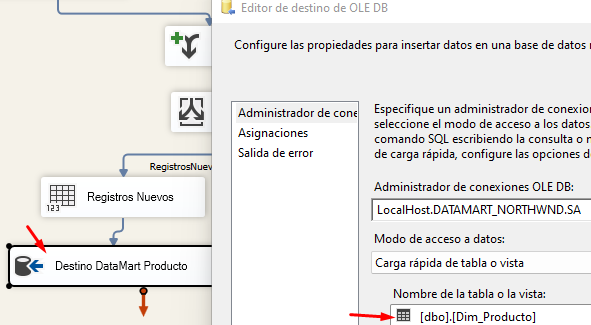
### División Condicional

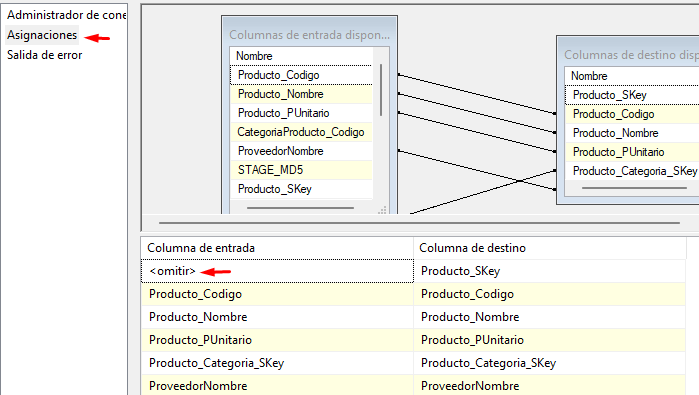


### Contar Filas

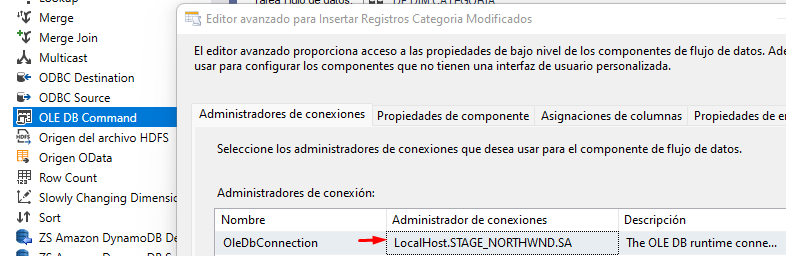


### Destino Data – Mart Cliente (nuevos)





### Comando OLDB Insertar Registros Modificados



CREATE TABLE [STAGE\_NORTHWND].[dbo].[Dim\_Cliente\_Mod](

[ClienteID] [int] IDENTITY(1,1) NOT NULL,

[Cliente\_Codigo] [char](5) NOT NULL,

[Cliente\_Nombre] [varchar](40) NOT NULL,

[Cliente\_Compania] [varchar](40) NULL,

[Cliente\_Direccion] [varchar](60) NULL,

[Cliente\_Ciudad] [varchar](15) NULL,

[Cliente\_Region] [varchar](25) NULL,

[Cliente\_Pais] [varchar](15) NULL,

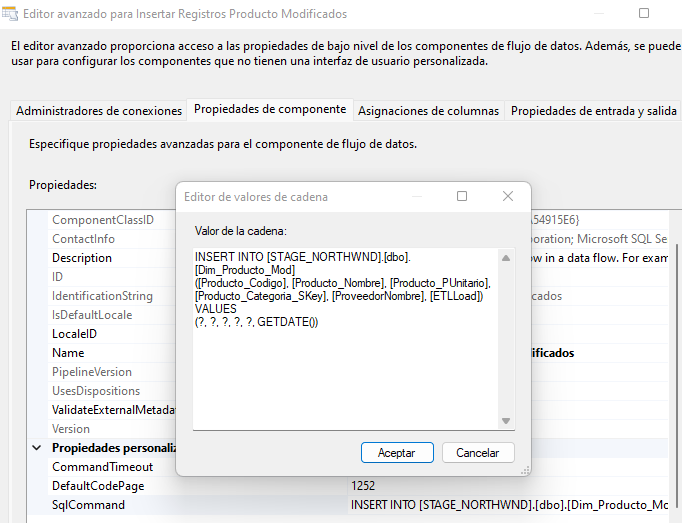
[ETLLoad] [DATETIME] )

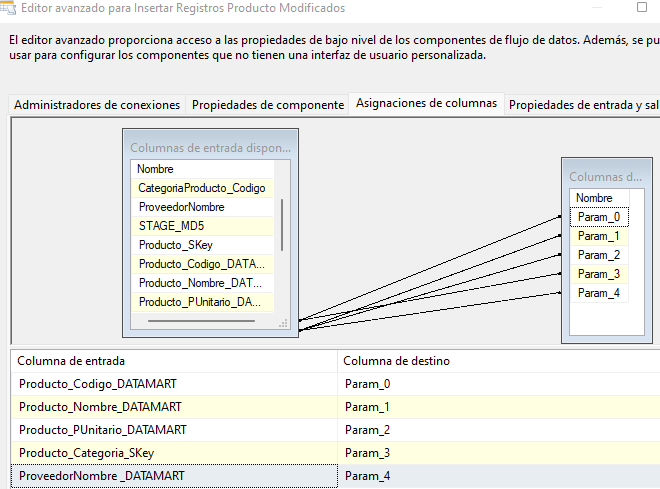
INSERT INTO [STAGE\_NORTHWND].[dbo].[Dim\_Cliente\_Mod]

([Cliente\_Codigo], [Cliente\_Nombre], [Cliente\_Compania], [Cliente\_Direccion], [Cliente\_Ciudad], [Cliente\_Region], [Cliente\_Pais], [ETLLoad])

VALUES

(?, ?, ?, ?, ?, ?, ?, GETDATE())





### Comando OLDB Actualizar Registros Modificados

UPDATE M

SET

M.Cliente\_Nombre = S.Cliente\_Nombre,

M.Cliente\_Compania = S.Cliente\_Compania,

M.Cliente\_Direccion = S.Cliente\_Direccion,

M.Cliente\_Ciudad = S.Cliente\_Ciudad,

M.Cliente\_Region = S.Cliente\_Region,

M.Cliente\_Pais = S.Cliente\_Pais

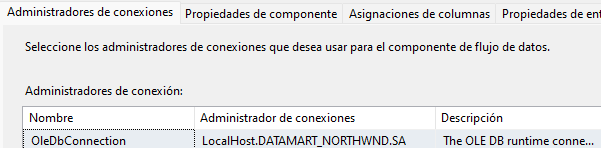
FROM [DATAMART\_NORTHWND].[dbo].[Dim\_Cliente] AS M

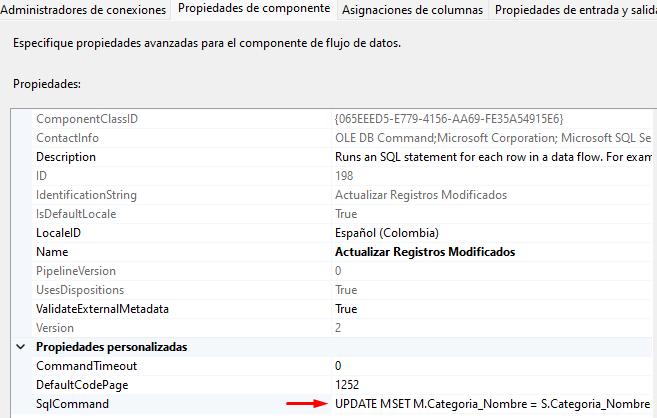
JOIN

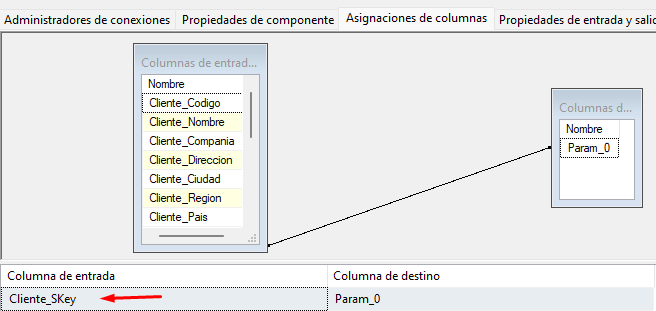
[STAGE\_NORTHWND].[dbo].[Stage\_Cliente] AS S

ON M.Cliente\_Codigo = S.Cliente\_Codigo

WHERE M.Cliente\_SKey = ?





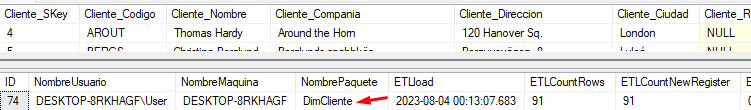


Prueba

SELECT \* FROM [DATAMART\_NORTHWND].[dbo].[Dim\_Cliente]

SELECT \* FROM [NORTHWND\_METADATA].[dbo].[ETLExecution] order by 1 desc

SELECT \* FROM [STAGE\_NORTHWND].[dbo].[Dim\_Cliente\_Mod]



# IMPLEMENTAR HECHOS FACT-VENTAS

# 

# ORIGEN DATA-MART FACT VENTAS

SELECT Cliente\_SKey,

Empleado\_SKey,

Producto\_SKey,

Tiempo\_SKey,

Ventas\_NOrden,

Ventas\_Monto,

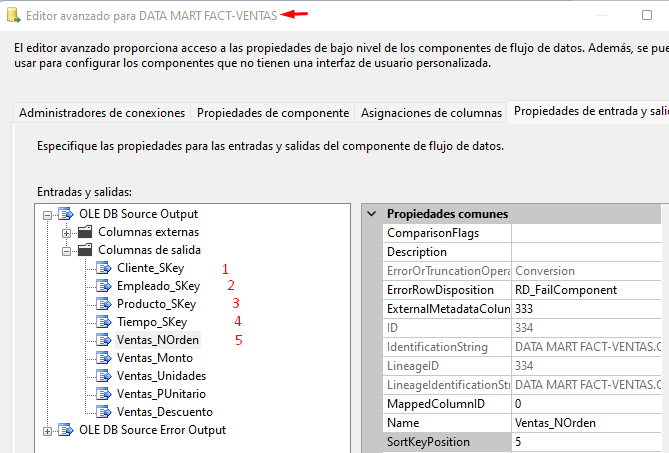
Ventas\_Unidades,

Ventas\_PUnitario,

Ventas\_Descuento

FROM [DATAMART\_NORTHWND].[dbo].[Fact\_Ventas]

ORDER BY Cliente\_SKey, Empleado\_SKey, Producto\_SKey, Tiempo\_SKey, Ventas\_NOrden



## ORIGEN STAGE DIM VENTAS

SELECT MC.Cliente\_SKey

,ME.Empleado\_SKey

,MP.Producto\_SKey

,MT.Tiempo\_Skey

,SV.[Ventas\_NOrden]

,SV.[Ventas\_Monto]

,SV.[Ventas\_Unidades]

,SV.[Ventas\_PUnitario]

,SV.[Ventas\_Descuento]

FROM [STAGE\_NORTHWND].[dbo].[Stage\_Ventas] AS SV

JOIN [DATAMART\_NORTHWND].[dbo].[Dim\_Cliente] AS MC

ON SV.Cliente\_Codigo = MC.Cliente\_Codigo

JOIN [DATAMART\_NORTHWND].[dbo].[Dim\_Empleado] AS ME

ON SV.Empleado\_Codigo = ME.Empleado\_Codigo

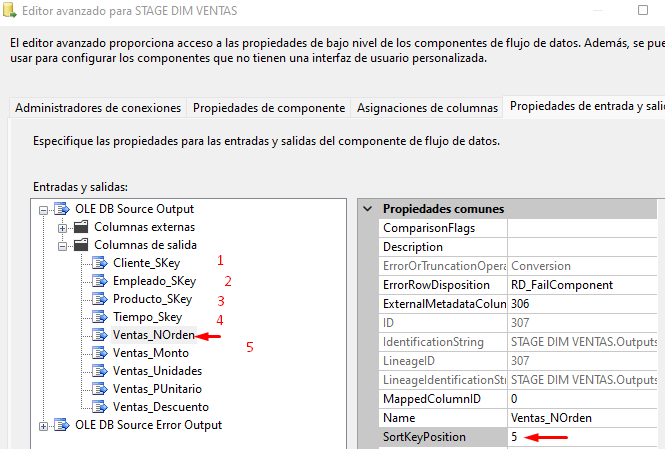
JOIN [DATAMART\_NORTHWND].[dbo].[Dim\_Producto] AS MP

ON SV.Producto\_Codigo = MP.Producto\_Codigo

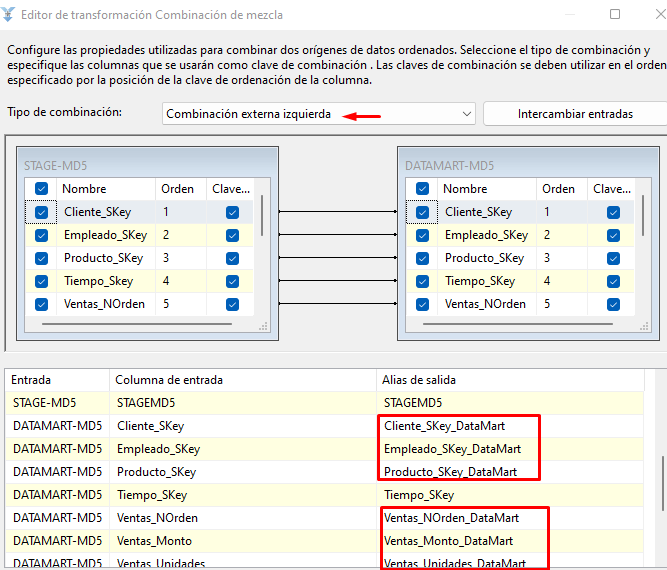
JOIN [DATAMART\_NORTHWND].[dbo].[Dim\_Tiempo] as MT

ON MT.Tiempo\_FechaActual = SV.Ventas\_OrderDate

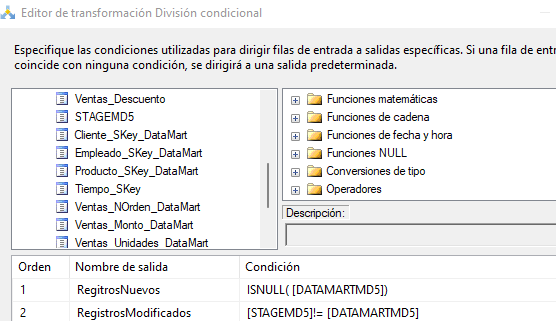
ORDER BY MC.Cliente\_SKey, ME.Empleado\_SKey, MP.Producto\_SKey, MT.Tiempo\_Skey, SV.[Ventas\_NOrden]



## COMBINACION DE MEZCLA



## División condicional



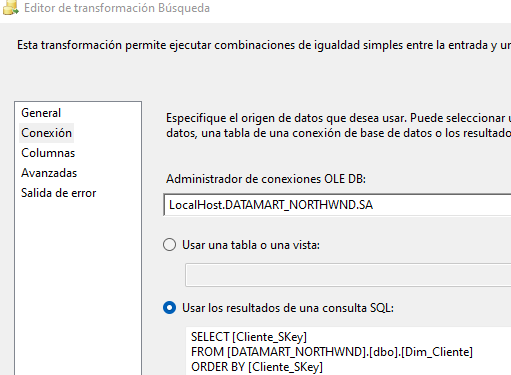
## BUSQUEDAS

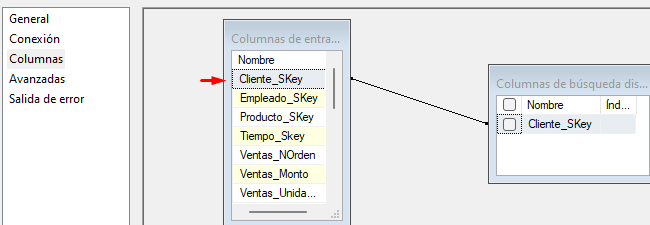
### CLIENTES COINCIDENTES

SELECT [Cliente\_SKey]

FROM [DATAMART\_NORTHWND].[dbo].[Dim\_Cliente]

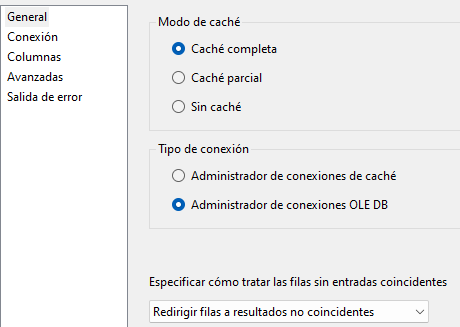
ORDER BY [Cliente\_SKey]

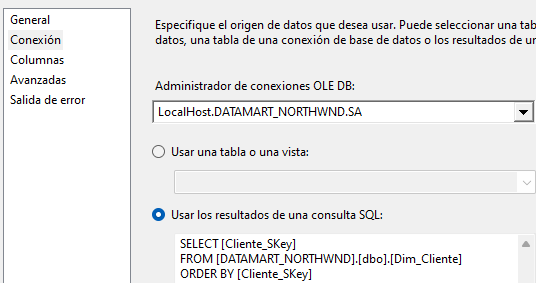




### CLIENTES NO COINCIDENTES

Misma consulta de la anterior



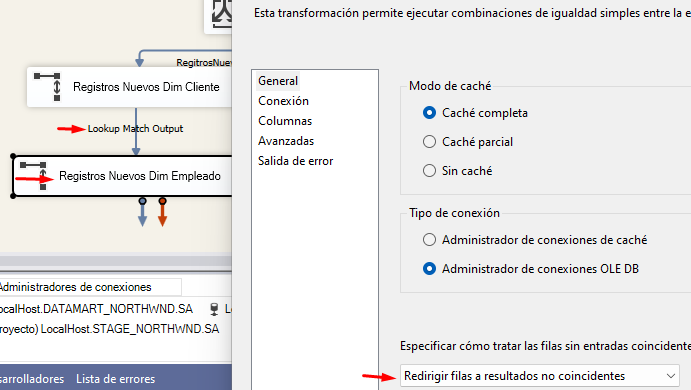


### EMPLEADOS COINCIDENTES

SELECT [Empleado\_SKey]

FROM [DATAMART\_NORTHWND].[dbo].[Dim\_Empleado]

ORDER BY [Empleado\_SKey]

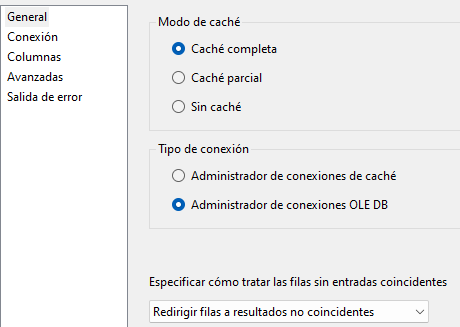


### 

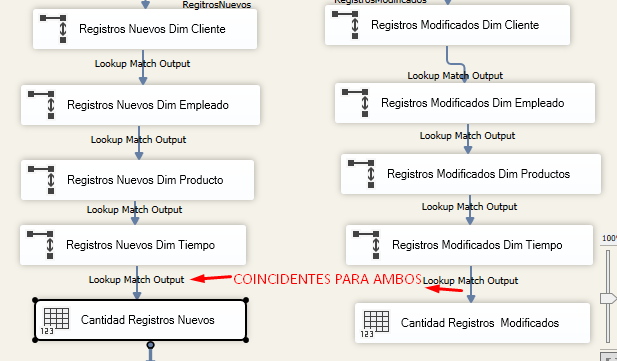
### 

### EMPLEADOS NO COINCIDENTES

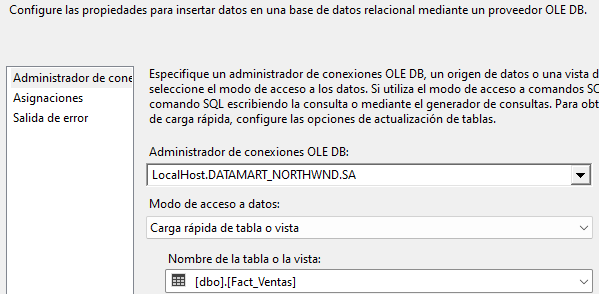
Misma consulta de la anterior



### CONTEO DE FILAS



### DESTINO FAC-VENTAS DE HECHOS (LA MITAD)



### COMAND OLDB INSERTAR FAC-VENTAS DE HECHOS MOD

CREATE TABLE [STAGE\_NORTHWND].[dbo].[Fact\_Ventas\_Mod](

[ID] [int] IDENTITY(1,1) NOT NULL,

[Cliente\_SKey] [int] NOT NULL,

[Empleado\_SKey] [int] NOT NULL,

[Producto\_SKey] [int] NOT NULL,

[Tiempo\_Skey] [int] NOT NULL,

[Ventas\_NOrden] [int] NOT NULL,

[Ventas\_Monto] [decimal](15, 2) NOT NULL,

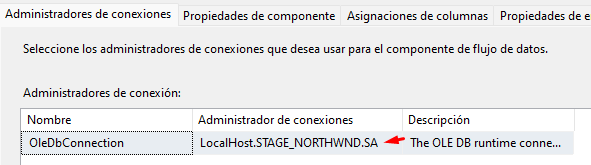
[Ventas\_Unidades] [int] NOT NULL,

[Ventas\_PUnitario] [decimal](15, 2) NOT NULL,

[Ventas\_Descuento] [decimal](15, 2) NOT NULL,

[ETLLoad] [DATETIME] )

Establecer conexión

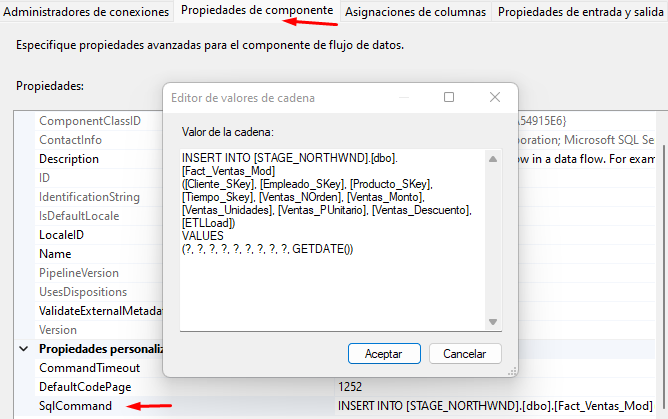


INSERT INTO [STAGE\_NORTHWND].[dbo].[Fact\_Ventas\_Mod]

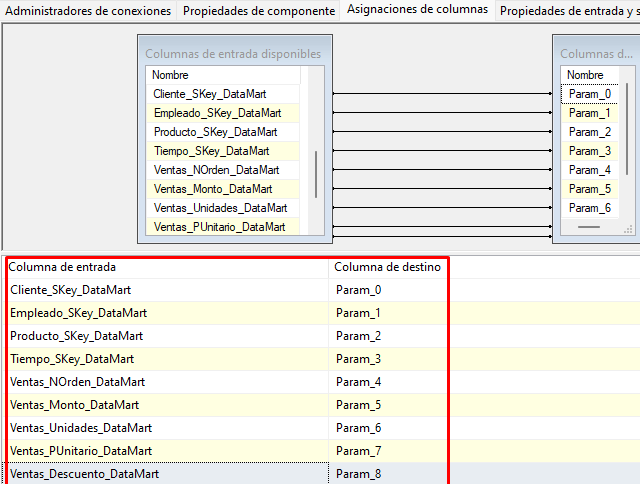
([Cliente\_SKey], [Empleado\_SKey], [Producto\_SKey], [Tiempo\_Skey], [Ventas\_NOrden], [Ventas\_Monto], [Ventas\_Unidades], [Ventas\_PUnitario], [Ventas\_Descuento], [ETLLoad])

VALUES

(?, ?, ?, ?, ?, ?, ?, ?, ?, GETDATE())

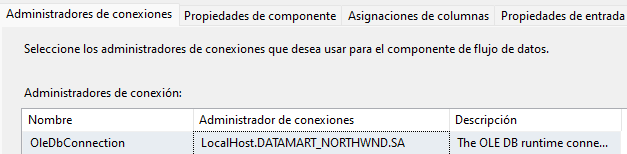


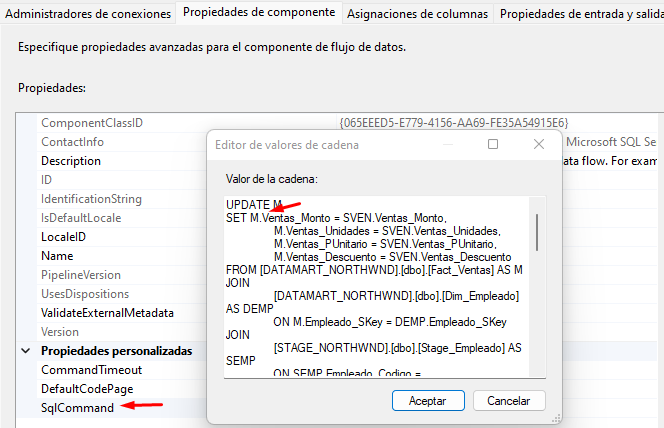
Establecer Parámetros

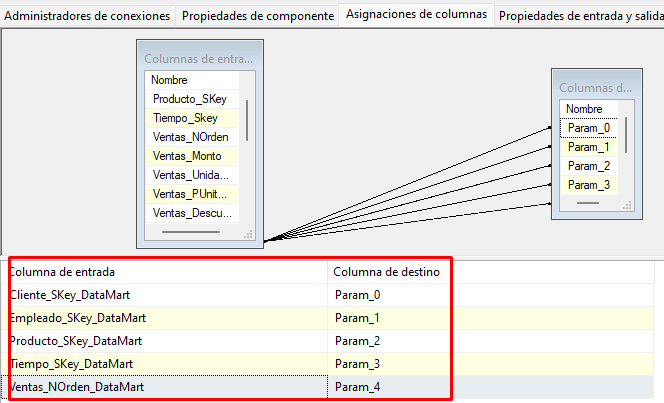


### COMAND OLDB INSERTAR DATAMART FAC-VENTAS DE HECHOS

Establecer conexión







SELECT \* FROM Fact\_Ventas

SELECT \* FROM [NORTHWND\_METADATA].[dbo].[ETLExecution] order by 1 desc

SELECT \* FROM [STAGE\_NORTHWND].[dbo].[Fact\_Ventas\_Mod]

