



Universidad Don Bosco
Consagrar la vida a la verdad

**UNIVERSIDAD DON BOSCO
FACULTAD DE INGENIERIA
DATAWAREHOUSE Y
MINERÍA DE DATOS
DMD941 G01T CICLO 01-23**

ACTIVIDAD:

DESAFIO 1:

CATEDRÁTICO:

ING. HERSON MIGUEL SERRANO
CHACÓN

ESTUDIANTES:

DRIOTIS CRUZ, DAVID OTONIEL

FLORES QUINTANILLA, ROBERTO CARLOS

CARNÉ:

DC211280

FQ211776

PORCENTAJE DE CUMPLIMIENTO

1- SPA DIEGO

100%

2- FIORELLA

100%

3- FIODIO

0%

CUIDADELA DON BOSCO, 07 DE MARZO DE 2023

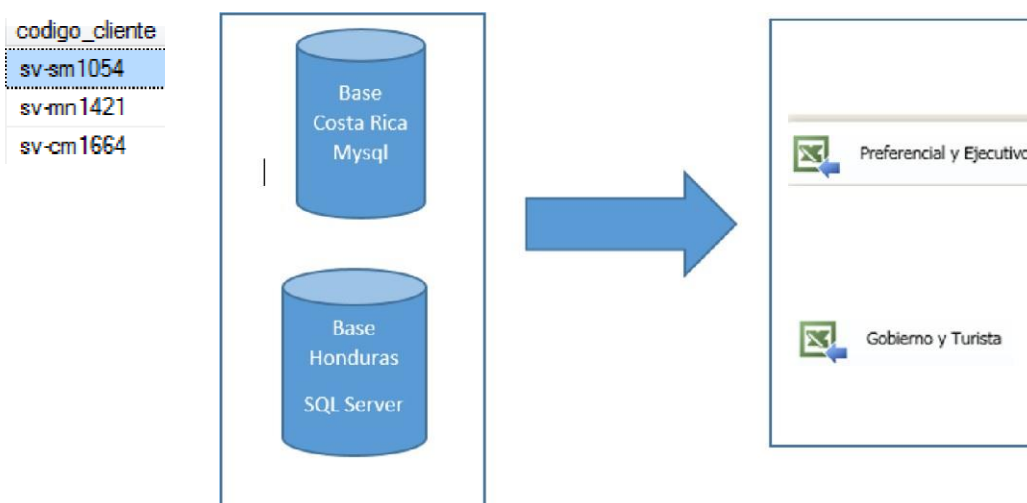
Indicaciones:

- ❑ El desafío puede ser en pareja o individual.
- ❑ Se debe hacer un documento con una portada con los integrantes, donde se haga las capturas de pantallas del funcionamiento del proceso y el porcentaje que se alcanzó (100% , 80% , etc.)
- ❑ El desarrollo del desafío y el documento en formato pdf, se debe compartir en aula digital en **un enlace público de GIT**.
- ❑ Todas las dudas serán **ATENDIDAS** por medio de **Discord** (para ayudarnos todos)
- ❑ Propuestas similares, serán sancionados ambos grupos con el 50% de la nota.

Ejercicios:

Para los dos primeros ejercicios se comparten archivos csv, para el tercero son archivos sql, para ser cargados por medio de un ETL en la base de datos y ser analizados, se deben de compartir las consultas sql hechas para análisis de cada problema expuesto.

1. **(40%)** El Spa, "**Diego**", necesita segmentar sus clientes, para realizar una campaña de fidelización, y le pide a usted que efectué un análisis de sus tres sucursales, que defina cuantos grupos y que características tienen.
2. **(40%)** La Floristería "**Fiorella**" quiere saber cómo se compran sus productos, y tiene la data de tres departamentos del país, por lo cual les pide su opinión sobre qué productos sobresalen, que combinaciones son mejores y quieren este estudio por departamento y también por país.
3. **(20%)** La telefonía "**FioDio**" solicita realizar un ETL que exporte una base de datos de Mysql y SQL Server, al final el destino serán dos archivos de Excel en donde en un archivo estarán los clientes preferenciales y ejecutivos y en el segundo los de gobierno y turista, adicional en los archivos de Excel se deberá crear un campo código de país, que se llenará sustrayendo los dos primeros caracteres de código cliente, ver imagen a continuación.

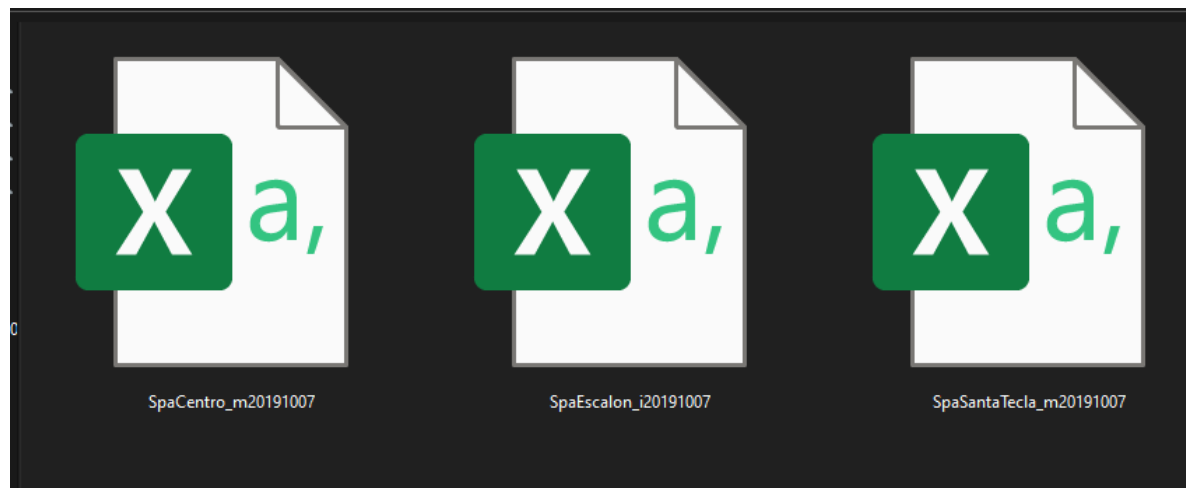


Ejercicio 1 - Spa Diego

Para la realización de este ejercicio contamos con los datos de 3 sucursales

1. Sucursal Centro
2. Sucursal Escalón
3. Sucursal Santa Tecla

La información de cada sucursal nos es proveída en archivos con extensión .csv, a razón de uno por cada sucursal:



Siendo esta la estructura que presenta cada archivo que representa una sucursal.

	A	B	C	D	E	F	G	H	I	
1	id	Sexo	ingresos	PromVisit	Edad	Sauna	Masaje	Hidro	Yoga	
2	Tomkin Stickles	1	2555.23	1.94	21	1	1	0	0	
3	Tyson Stovine	1	2476.87	5.83	29	1	1	0	1	
4	Miller Carnachen	1	1209.36	6.07	23	1	0	0	0	
5	Darnell Dine-Hart	1	1307.02	3.17	63	1	1	1	1	
6	Wyatt Keyte	1	1511.78	2.08	41	0	0	0	0	
7	Trip Vost	1	772.08	1.74	52	1	1	0	0	
8	Ammamaria D'Errico	0	2749.35	4.58	41	0	0	1	1	
9	Jessica Kuhn	1	1964.11	2.3	51	1	1	1	0	
10	Hamnet Lindenberg	0	1577.93	2.44	62	1	1	1	0	
11	Marga Kiley	0	1945.16	6.36	25	1	1	0	1	
12	Pebrook Praton	1	1917.15	5.86	37	1	1	0	0	
13	Koenraad Marchiso	1	699.14	2.64	34	1	0	1	0	
14	Chase McManus	1	1945.28	2.52	47	0	1	1	1	
15	Robinett Sauvan	1	2065.14	4.78	43	1	1	0	0	
16	Jessica Dudding	1	988.99	3.72	47	1	1	0	1	

Es requerido segmentar los clientes, para realizar una campaña de fidelización, realizando un análisis de sus tres sucursales, que defina cuantos grupos y que características tienen.

Para ello buscaremos brindar respuesta a las siguientes preguntas:

- ✓ Promedio de ingresos según el sexo del cliente
- ✓ Promedio de visitas según el sexo del cliente
- ✓ La suma de cada servicio del spa
- ✓ Promedio de las edades de los clientes según su genero

Para resolver este problema procedemos a crear una base de datos llamada “EJERCICIO1” y dentro de ella una tabla llamada “Spa” para almacenar nuestros datos consolidados de las tres sucursales y poder el análisis de la información

Este es el encabezado de nuestro archivo Análisis_Diego.sql

```
-- -----
--      Proyecto:      DESAFIO 01 - DMD941
--      Ejercicio 1:    SPA DIEGO (40%)
--                      El Spa, "Diego", necesita segmentar sus clientes, para
--                      realizar una campaña de fidelización, y le pide a usted que
--                      efectué un análisis de sus tres sucursales, que defina
--                      cuantos grupos y que características tienen.
--      Materia:       DATAWAREHOUSE Y MINERÍA DE DATOS
--      Archivo:        Analisis_Diego.sql
--      Descripción:    Ingreso de datos ejemplo para base de datos
--      Alumnos:        Driotis Cruz, David Otoniel.....DC211280
--                      Flores Quintanilla, Roberto Carlos.....FQ211776
--      Código fuente:  https://github.com/DMD941/Desafio_01
-- -----
```

Con este script hacemos un DROP de la base de datos si ya existía anteriormente de una forma correcta, cerrando cualquier conexión a ella que este activa.

```
--Remove backup history
EXEC msdb.dbo.sp_delete_database_backuphistory @database_name =
N'EJERCICIO1'
GO
USE [master]
GO
--Cerramos todas las conexiones a la base de datos
ALTER DATABASE [EJERCICIO1] SET SINGLE_USER WITH ROLLBACK IMMEDIATE
GO
USE [master]
GO
--removemos la base de datos, si esta existe previamente
DROP DATABASE IF EXISTS [EJERCICIO1]
GO
```

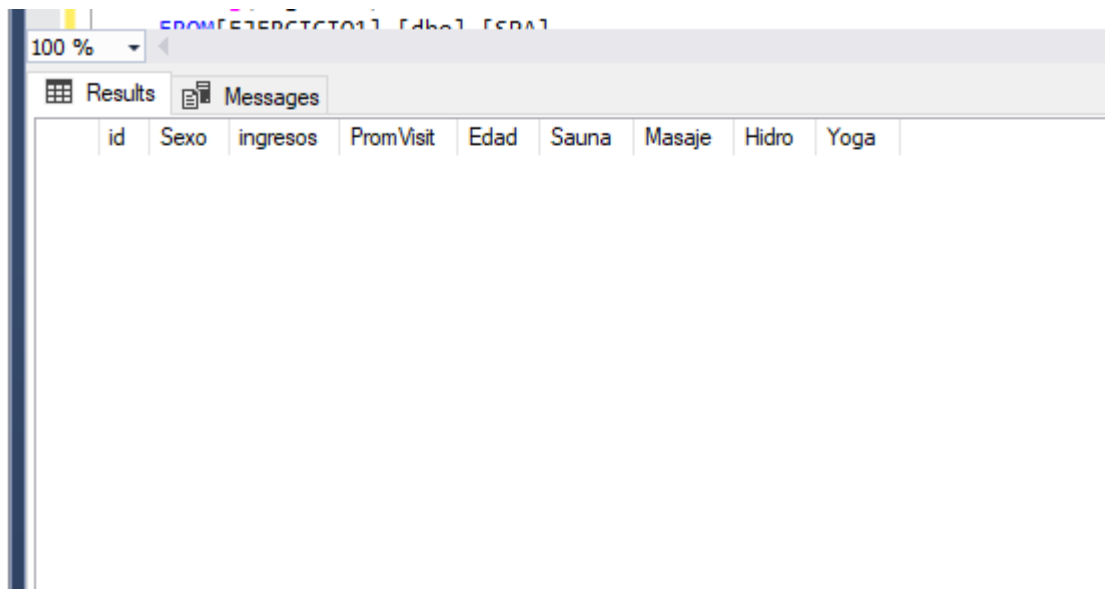
Procedemos a crear nuestra base de datos EJERCICIO1 y creamos la tabla donde almacenaremos los datos consolidados, para poder realizar nuestras consultas

```
--Creamos la base de datos EJERCICIO1
USE master;
CREATE DATABASE [EJERCICIO1]
GO
--Removemos la tabla SPA, si existiera previamente
DROP TABLE IF EXISTS [EJERCICIO1].[dbo].[SPA]
GO
--Creamos la tabla SPA
CREATE TABLE [EJERCICIO1].[dbo].[SPA] (
    [id] varchar(50),
    [Sexo] INT,
    [ingresos] DECIMAL,
    [PromVisit] DECIMAL,
    [Edad] INT,
    [Sauna] INT,
    [Masaje] INT,
    [Hidro] INT,
    [Yoga] INT
)
GO
```

Verificamos que la tabla fue creada al mostrar los primeros 1000 registros

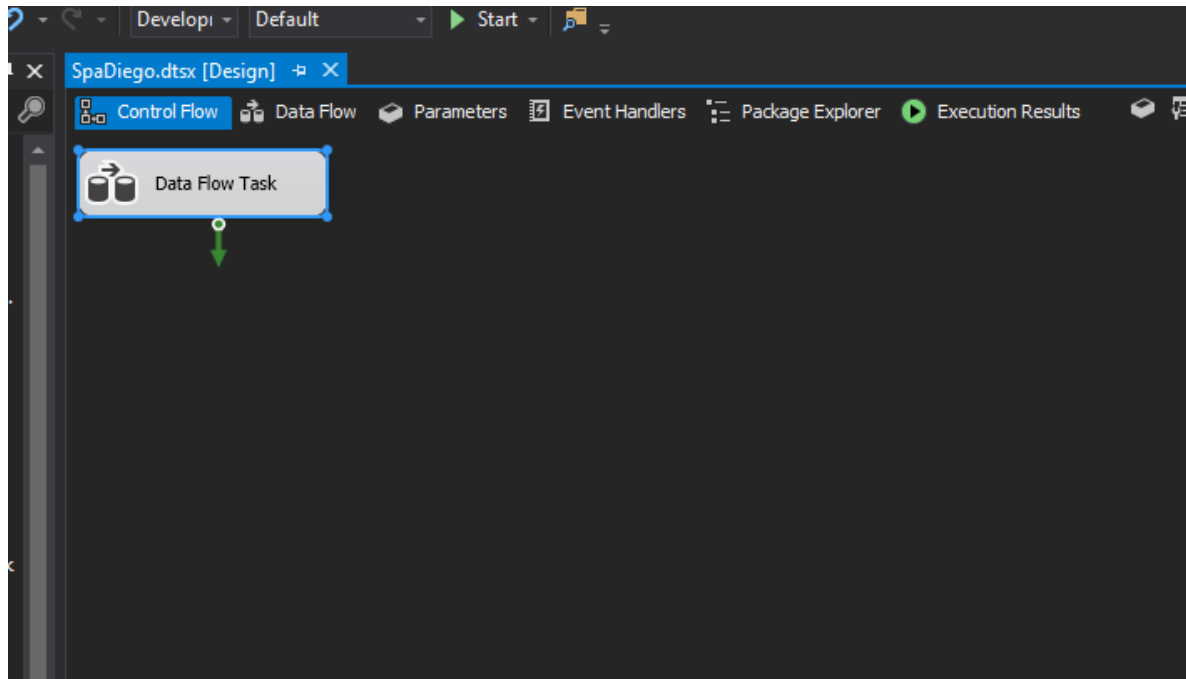
```
--Verificamos que la tabla fue creada al mostrar los primeros 1000
registros
SELECT TOP (1000) [id]
    , [Sexo]
    , [ingresos]
    , [PromVisit]
    , [Edad]
    , [Sauna]
    , [Masaje]
    , [Hidro]
    , [Yoga]
FROM [EJERCICIO1].[dbo].[SPA]
```

Podemos ver que nuestra tabla esta vacía:

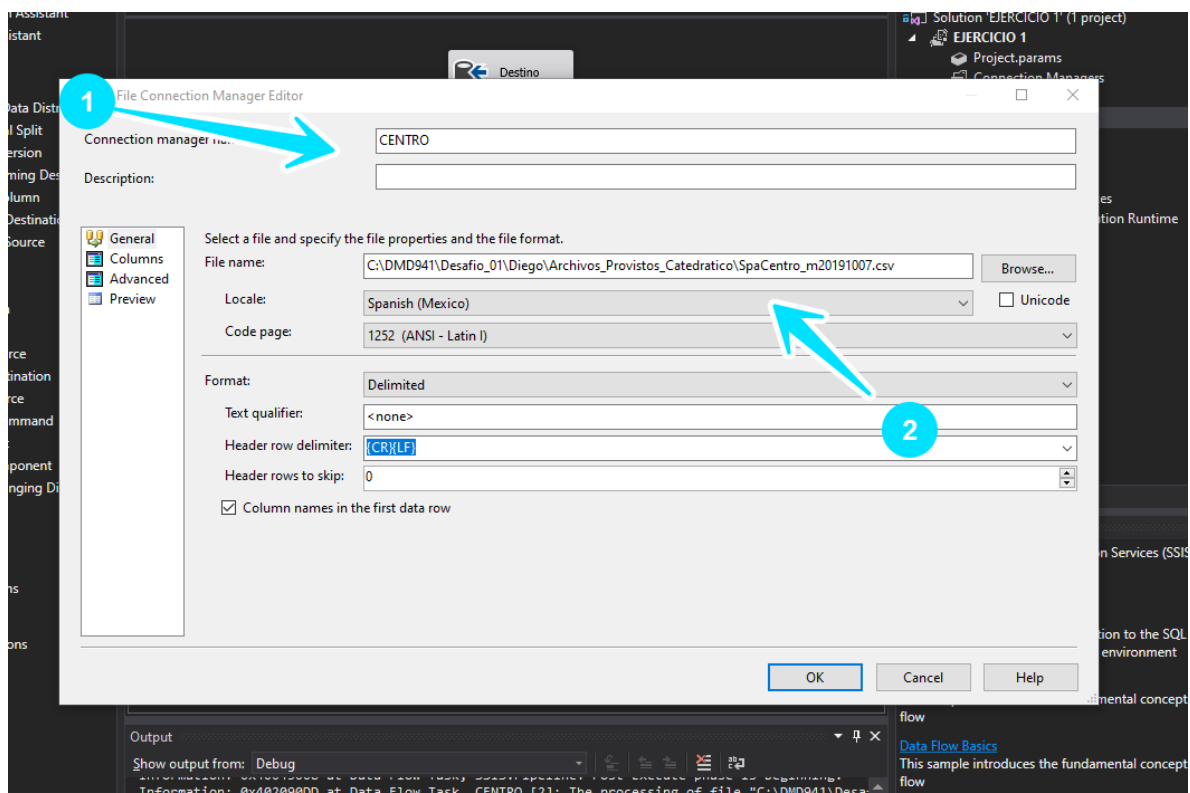


id	Sexo	ingresos	PromVisit	Edad	Sauna	Masaje	Hidro	Yoga
----	------	----------	-----------	------	-------	--------	-------	------

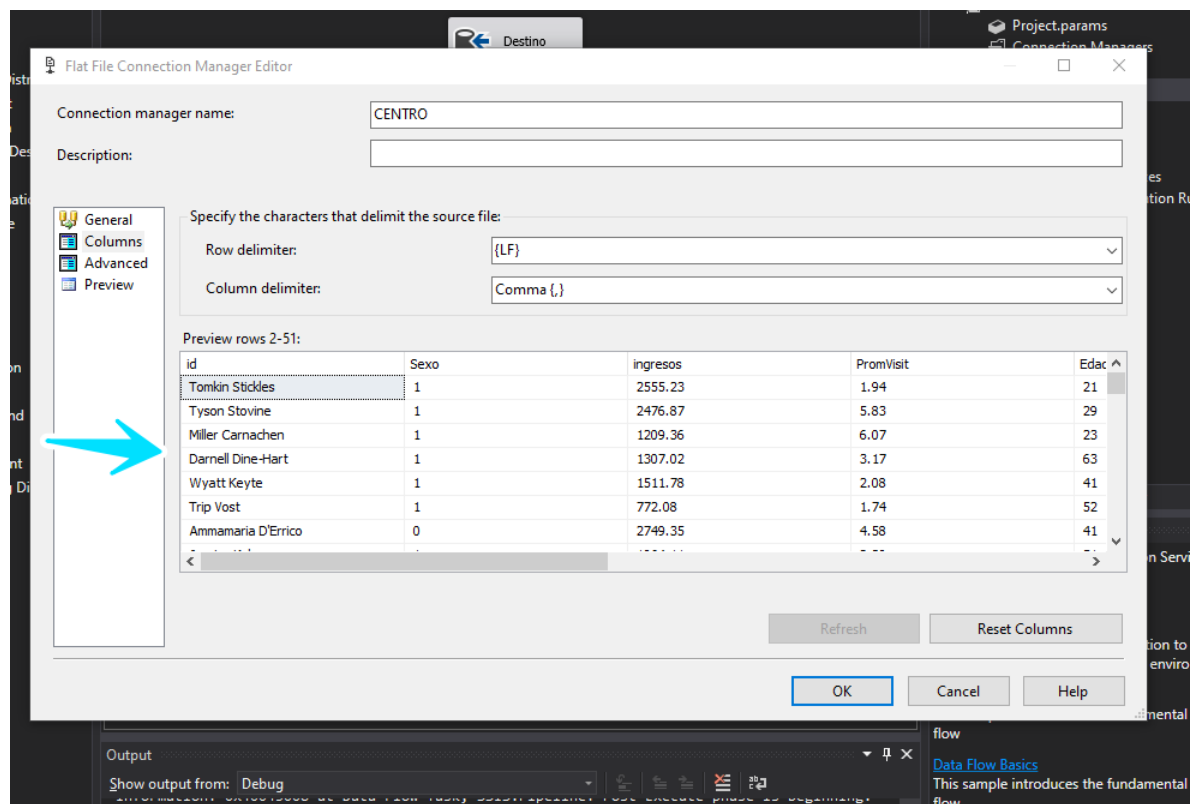
Procedemos a crear en SQL Server integration services dentro de nuestro Visual Studio nuestro Flow de datos



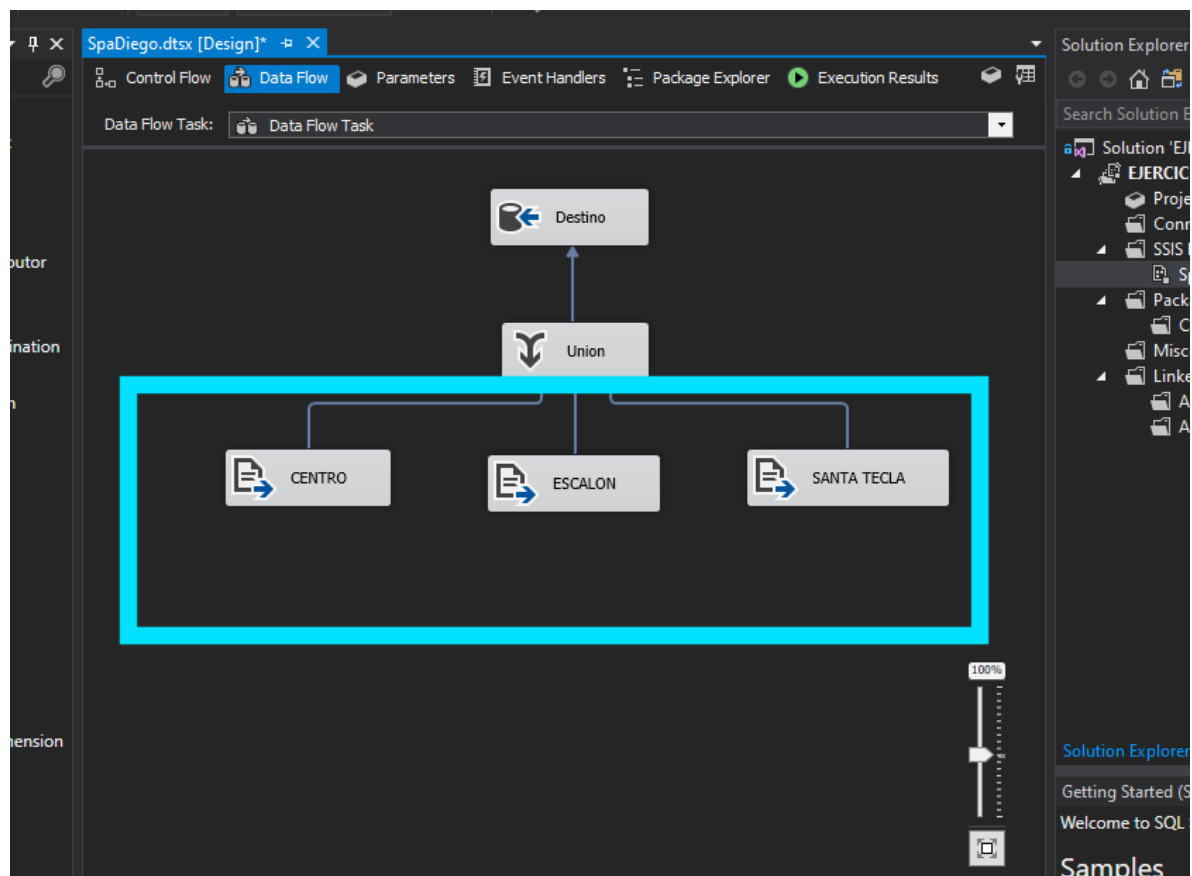
1. Procedemos a nombrar nuestra conexión a nuestro archivo plano
2. Marcamos la ruta a nuestro archivo plano de origen



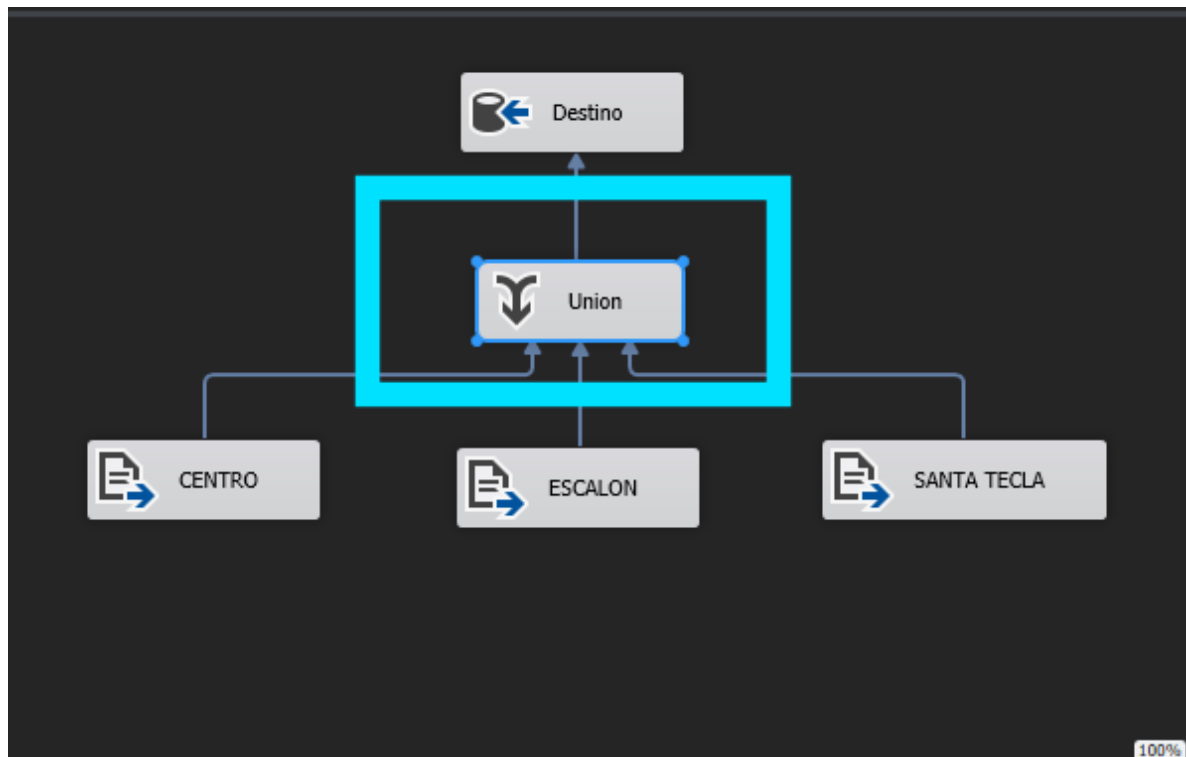
En esta parte podemos ver una vista previa de los datos.



Hacemos eso por cada una de las sucursales.



Configuramos como se llevara a cabo la unión de los registros



Union All Transformation Editor

Configure the properties used to merge multiple inputs into one output by creating mappings between columns.

Output Column Name	Union All Input 2	Union All Input 3	Union All Input 4
id	id	id	id
Sexo	Sexo	Sexo	Sexo
ingresos	ingresos	ingresos	ingresos
PromVisit	PromVisit	PromVisit	PromVisit
Edad	Edad	Edad	Edad
Sauna	Sauna	Sauna	Sauna
Masaje	Masaje	Masaje	Masaje
Hidro	Hidro	Hidro	Hidro
Yoga	Yoga	Yoga	Yoga

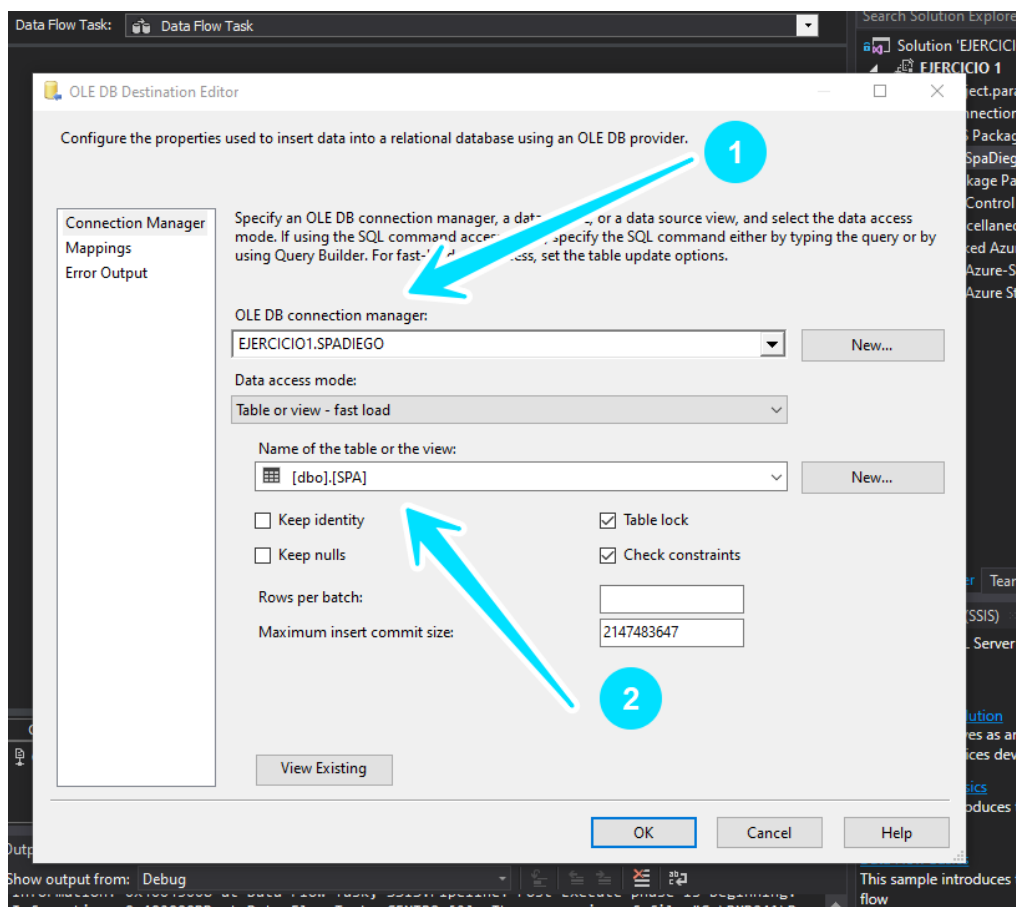
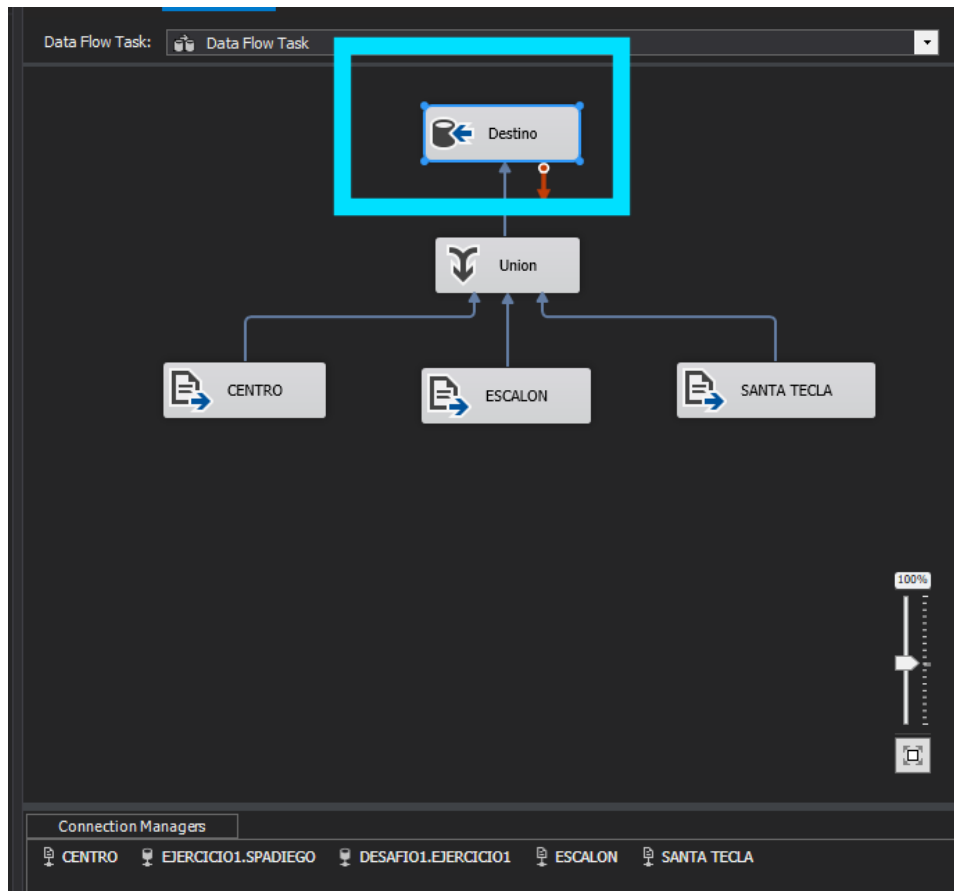
OK Cancel Help

Show output from: Debug

Information: 0x402090DD at Data Flow Task, CENTRO [2]: The processing of file "C:\DMD941\Desa-
Information: 0x402090DD at Data Flow Task, ESCALON [82]: The processing of file "C:\DMD941\De-
Information: 0x402090DD at Data Flow Task, SANTA TECLA [137]: The processing of file "C:\DMD941\De-

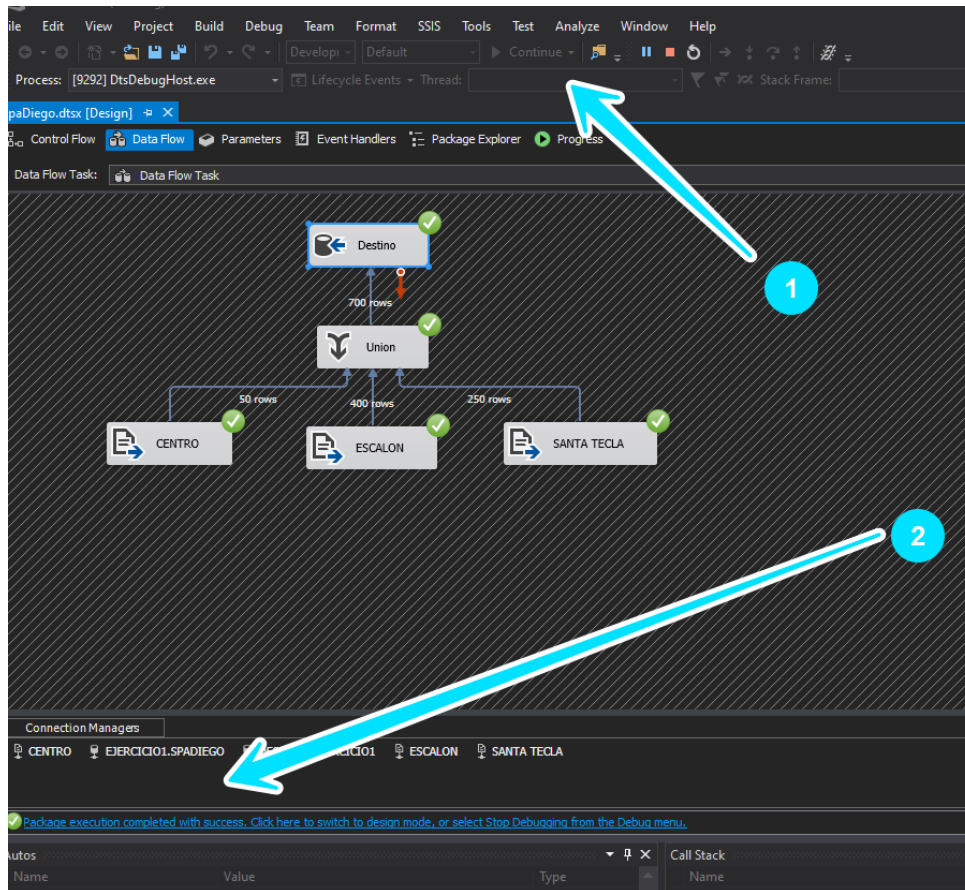
This sample introduces the fun
flow

En nuestro OLE DB destination definimos la tabla donde se vaciaran los datos



Procedemos a:

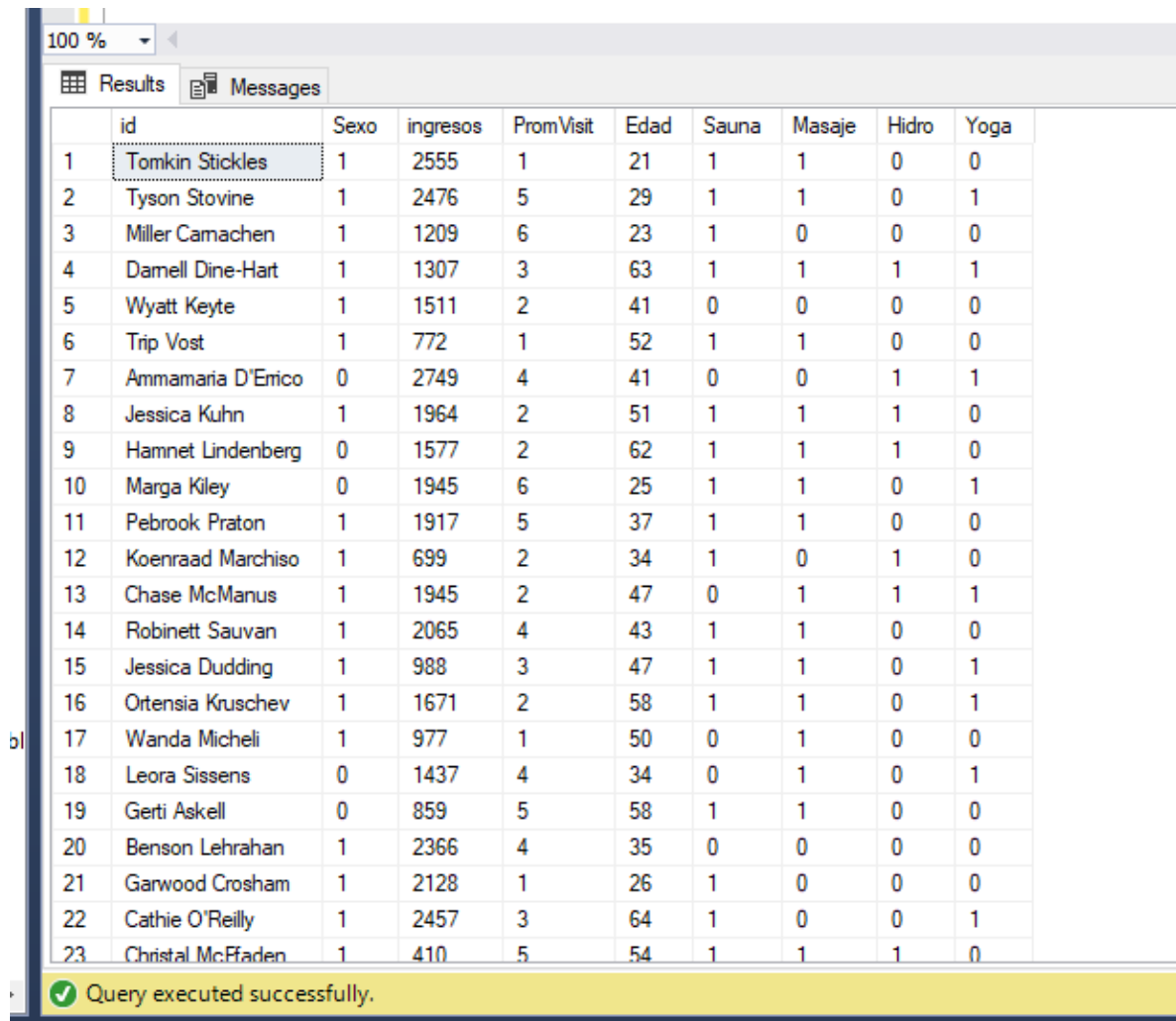
1. Ejecutar nuestro proyecto
2. Corroborar que se ha efectuado correctamente



Ahora en **SQL Server Management Studio** Verificamos que la tabla fue actualizada al mostrar los primeros 1000 registros

--Verificamos que la tabla fue creada al mostrar los primeros 1000 registros

```
SELECT TOP (1000) [id]
      ,[Sexo]
      ,[ingresos]
      ,[PromVisit]
      ,[Edad]
      ,[Sauna]
      ,[Masaje]
      ,[Hidro]
      ,[Yoga]
FROM [EJERCICIO1].[dbo].[SPA]
```



	id	Sexo	ingresos	PromVisit	Edad	Sauna	Masaje	Hidro	Yoga
1	Tomkin Stickles	1	2555	1	21	1	1	0	0
2	Tyson Stovine	1	2476	5	29	1	1	0	1
3	Miller Camachen	1	1209	6	23	1	0	0	0
4	Damell Dine-Hart	1	1307	3	63	1	1	1	1
5	Wyatt Keyte	1	1511	2	41	0	0	0	0
6	Trip Vost	1	772	1	52	1	1	0	0
7	Ammamaria D'Erico	0	2749	4	41	0	0	1	1
8	Jessica Kuhn	1	1964	2	51	1	1	1	0
9	Hamnet Lindenberg	0	1577	2	62	1	1	1	0
10	Marga Kiley	0	1945	6	25	1	1	0	1
11	Pebrook Praton	1	1917	5	37	1	1	0	0
12	Koenraad Marchiso	1	699	2	34	1	0	1	0
13	Chase McManus	1	1945	2	47	0	1	1	1
14	Robinett Sauvan	1	2065	4	43	1	1	0	0
15	Jessica Dudding	1	988	3	47	1	1	0	1
16	Ortensia Krushev	1	1671	2	58	1	1	0	1
17	Wanda Micheli	1	977	1	50	0	1	0	0
18	Leora Sissens	0	1437	4	34	0	1	0	1
19	Gerti Askill	0	859	5	58	1	1	0	0
20	Benson Lehrahan	1	2366	4	35	0	0	0	0
21	Garwood Crosham	1	2128	1	26	1	0	0	0
22	Cathie O'Reilly	1	2457	3	64	1	0	0	1
23	Christal McFaden	1	410	5	54	1	1	1	0

Query executed successfully.

Ahora procedemos a interpretar nuestros datos.

Primero analizaremos el promedio de ingresos según el genero del cliente

```
-- PROMEDIO DE INGRESOS SEGUN EL SEXO DEL CLIENTE

SELECT avg(ingresos) as Promedio
FROM[EJERCICIO1].[dbo].[SPA]
where Sexo = 0

SELECT avg(ingresos) as Promedio
FROM[EJERCICIO1].[dbo].[SPA]
where Sexo = 1
```

	Promedio
1	1654.199494

where sexo = 0

	Promedio
1	1711.963815

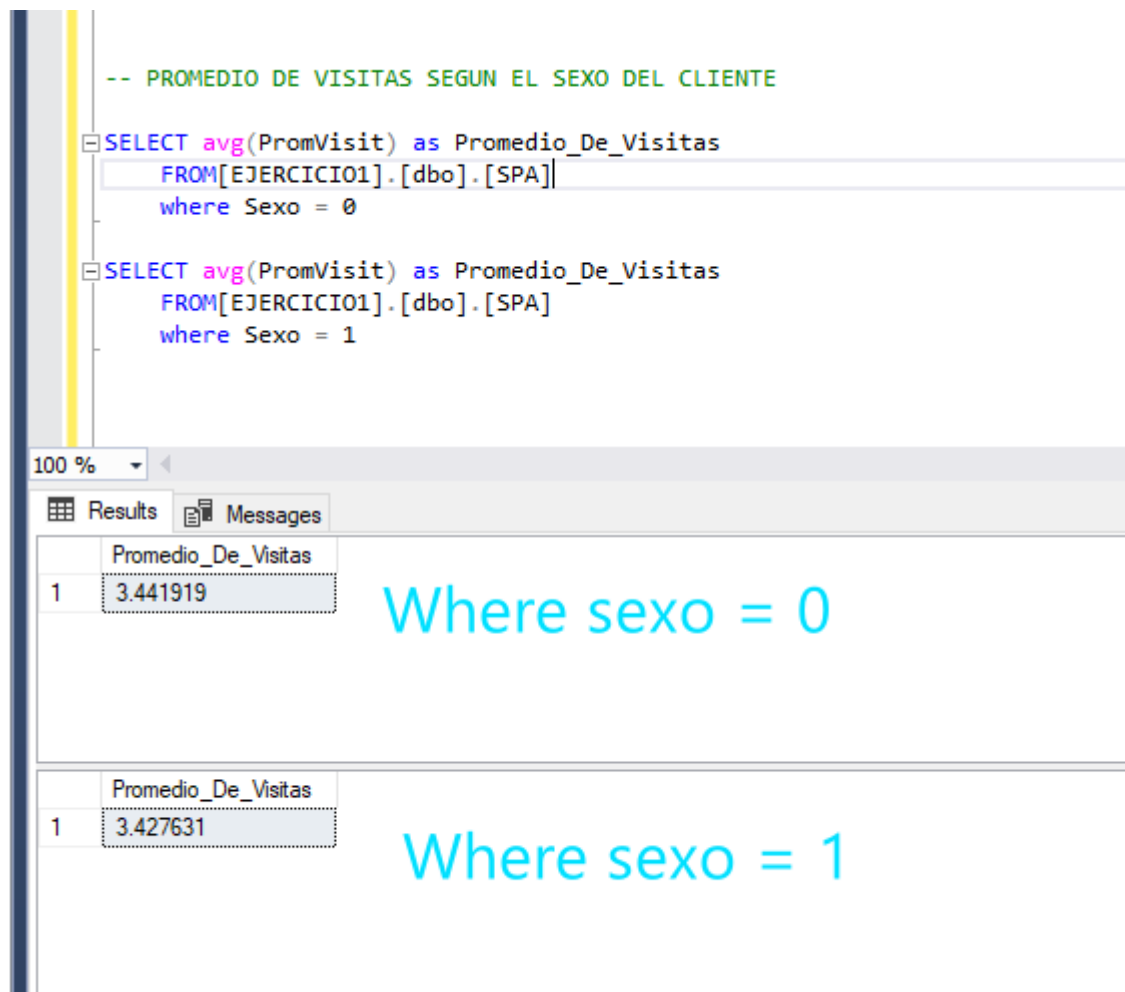
where sexo = 1

Concluimos que el promedio de ingresos por el sexo = 1 es mayor al generado por sexo = 0

Luego investigamos la cantidad de visitas según el genero del cliente

```
-- PROMEDIO DE VISITAS SEGUN EL SEXO DEL CLIENTE
SELECT avg(PromVisit) as Promedio_De_Visitas
FROM[EJERCICIO1].[dbo].[SPA]
where Sexo = 0

SELECT avg(PromVisit) as Promedio_De_Visitas
FROM[EJERCICIO1].[dbo].[SPA]
where Sexo = 1
```



The screenshot displays the SQL Server Enterprise Manager interface. The top pane shows two SQL queries. The first query filters for 'Sexo = 0' and the second for 'Sexo = 1'. The bottom pane shows the results of these queries. The first result set, for 'Where sexo = 0', shows a single row with the value 3.441919. The second result set, for 'Where sexo = 1', shows a single row with the value 3.427631. The interface includes a 'Results' tab and a 'Messages' tab, and a zoom level of 100% is indicated.

	Promedio_De_Visitas
1	3.441919

Where sexo = 0

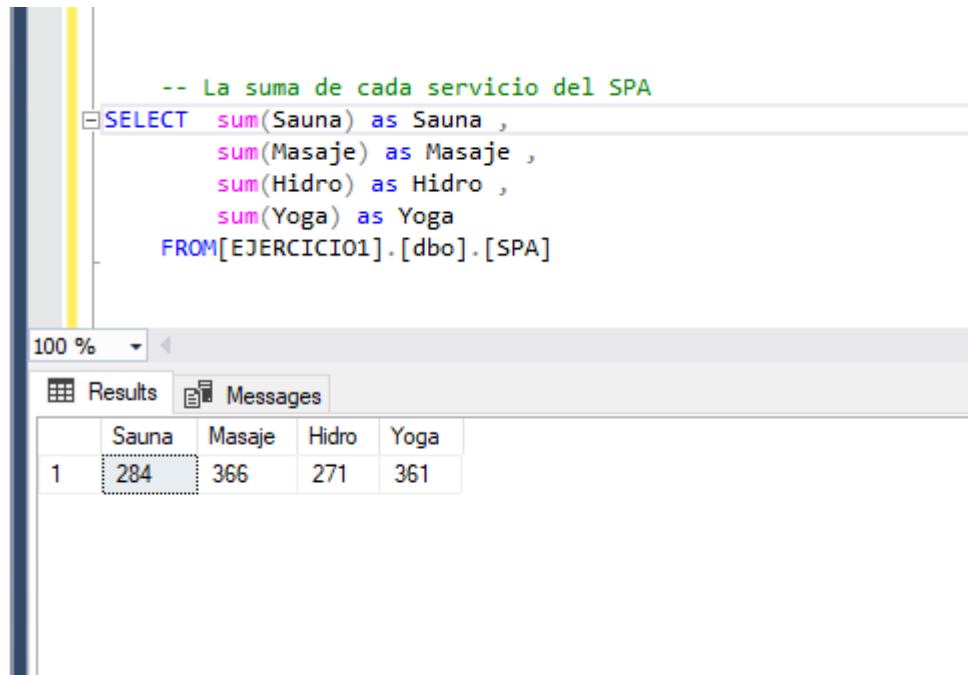
	Promedio_De_Visitas
1	3.427631

Where sexo = 1

Concluimos que los clientes de sexo = 0 visitan el spa de forma mas asidua, aunque la variación es mínima entre ambos sexos, mostrando un comportamiento similar.

Ahora vamos a investigar la suma de de cada servicio del spa

```
-- La suma de cada servicio del SPA
SELECT  sum(Sauna) as Sauna ,
        sum(Masaje) as Masaje ,
        sum(Hidro) as Hidro ,
        sum(Yoga) as Yoga
FROM[EJERCICIO1].[dbo].[SPA]
```



The screenshot shows the SQL Server Enterprise Manager interface. The query editor at the top contains the same SQL query as above. Below the editor, the 'Results' tab is active, displaying the query's output in a table format. The table has four columns: Sauna, Masaje, Hidro, and Yoga. The first row shows the sum for each service: Sauna (284), Masaje (366), Hidro (271), and Yoga (361).

	Sauna	Masaje	Hidro	Yoga
1	284	366	271	361

Concluimos que el servicio mas pedidos son los masajes, seguido del yoga, le sigue el sauna y por ultimo el menos demandado es el Hidro

Por ultimo vamos a promediar las edades de los clientes.

```
-- Promedio de las edades de los clientes según su genero
SELECT avg(Edad) as Edad
      FROM[EJERCICIO1].[dbo].[SPA]
      WHERE Sexo = 0

SELECT avg(Edad) as Edad
      FROM[EJERCICIO1].[dbo].[SPA]
      WHERE Sexo = 1
```

-- Promedio de las edades de los clientes segun su genero

SELECT avg(Edad) as Edad
FROM[EJERCICIO1].[dbo].[SPA]
WHERE Sexo = 0

SELECT avg(Edad) as Edad
FROM[EJERCICIO1].[dbo].[SPA]
WHERE Sexo = 1

100 %

Results Messages

	Edad
1	41

where sexo = 0

	Edad
1	43

where sexo = 1

Concluimos que la edad promedio de los clientes con sexo = 0 es de 41 años y de los clientes de sexo = 1 es 43 años

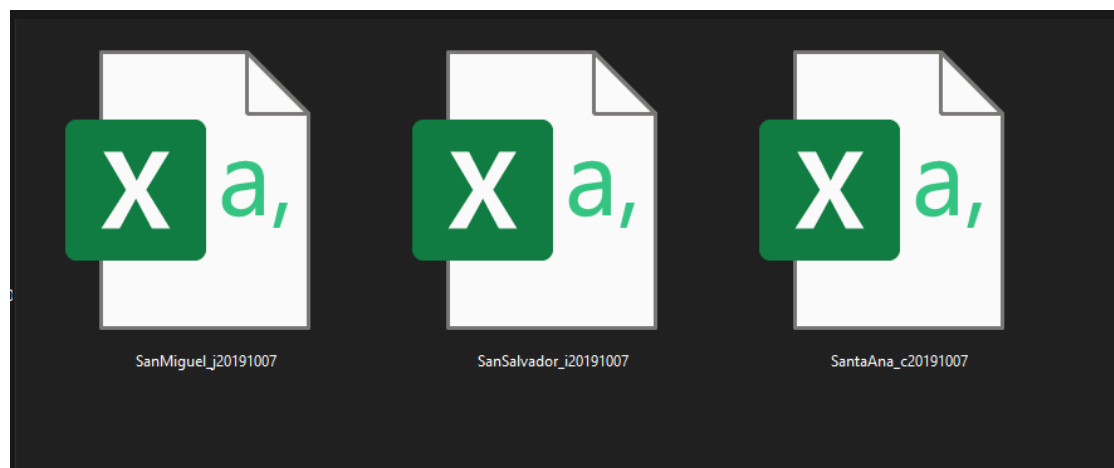
Ejercicio 2 - Spa Diego

La Floristería "Fiorella" quiere saber cómo se compran sus productos, y tiene la data de tres departamentos del país, por lo cual les pide su opinión sobre qué productos sobresalen, que combinaciones son mejores y quieren este estudio por departamento y también por país.

Para la realización de este ejercicio contamos con los datos de 3 departamentos

1. San Miguel
2. San Salvador
3. Santa Ana

La información de cada departamento nos es proveída en archivos con extensión .csv, a razón de uno por cada departamento:



Siendo esta la estructura que presenta cada archivo que representa una sucursal.

A1																
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	
1	id	Rosas	Claveles	Macetas	Tierra	Girasoles	Hortensia	Globos	Tarjetas	fOrquÃ-dias	CarmesÃ-	Lirios	Aurora	Tulipanes	ListÃ³n	
2	Egon Greenh	1	0	0	0	1	1	1	0	0	0	1	0	0		
3	Elita Borles	1	0	0	0	1	1	0	0	0	1	1	0	1		
4	Kingsly Yerre	1	1	0	0	1	1	1	1	1	1	1	1	0		
5	Graehme Do	1	0	1	1	0	0	1	0	0	0	1	0	1		
6	Wini Mclury	1	1	0	0	1	1	0	1	1	0	0	1	1		
7	Abigael Halli	1	1	0	1	0	0	0	1	1	0	1	1	1		
8	Eldon Parret	0	1	0	0	0	0	1	0	1	0	0	0	0		
9	Bernelle Col	0	0	1	0	1	1	0	1	1	0	1	1	0		
10	Cordellie Bec	0	0	0	0	0	1	0	0	1	1	0	1	1		
11	Jody Mewrci	0	1	1	1	0	1	1	0	1	0	1	0	0		
12	Gill Aisbett	1	1	1	0	0	1	0	1	0	1	0	0	1		
13	Jarrad Bayle	0	1	0	0	1	0	1	0	1	1	1	1	1		
14	Carolyn Oxer	1	1	0	1	0	0	1	1	0	0	1	1	0		
15	Alain Dodsle	1	1	0	1	1	0	1	0	0	0	1	1	1		
16	Kelly Bohma	1	1	0	0	1	0	0	1	1	1	1	1	0		
17	Didi Agnew	1	0	0	1	1	0	1	0	1	1	1	1	1		
18	Culver Simki	1	1	1	0	1	0	1	1	0	1	0	1	0		
19	Lenka Josuw	1	0	1	0	1	0	0	0	1	0	1	0	0		
20	Rochell Pitsc	1	1	0	0	0	0	0	0	0	0	1	0	1		
21	Laurie Harle	0	0	0	0	0	1	0	1	0	0	0	0	1		
22	Cassandry Er	0	0	0	0	0	1	0	0	1	1	0	1	1		
23	Sharleen Del	0	0	1	1	1	1	0	0	0	1	1	0	1		
24	Lindon Ansti	0	0	1	0	1	0	0	1	1	0	1	0	1		
25	Deeyn Feren	1	0	1	1	0	1	1	1	1	1	0	0	1		
26	Valenka Gou	0	0	0	1	0	1	0	0	1	0	0	1	1		
27	Darnall Giott	1	0	0	0	1	0	1	0	1	1	0	0	1		
28	Cristabel Gar	1	0	0	1	0	1	1	1	1	1	0	0	0		
29	Chalmers Go	1	0	0	1	0	1	1	1	1	0	1	0	1		

Es requerido segmentar los clientes, realizando un análisis de los tres departamentos y un consolidado a nivel nacional, que defina qué productos sobresalen, que combinaciones son mejores

Para ello buscaremos brindar respuesta a las siguientes preguntas:

- ✓ Ventas de cada producto por departamento
- ✓ Ventas de cada producto a nivel nacional
- ✓ Que productos sobresalen

Para resolver este problema procedemos a crear una base de datos llamada “EJERCICIO2” y dentro de ella tres tablas llamadas “Floristeria_Santa_Ana”, “Floristeria_San_Salvador”, “Floristeria_San_Miguel”, para almacenar nuestros datos por departamento, así como también crearemos la tabla “Floristeria_Pais” para realizar un consolidado de los tres departamentos poder el análisis de la información

Este es el encabezado de nuestro archivo Análisis_Fiorella.sql

```
-- -----  
-- Proyecto: DESAFIO 01 - DMD941  
-- Ejercicio 2: FLORISTERIA FIORELLA (40%)  
-- La Floristería "Fiorella" quiere saber cómo se  
-- compran sus productos, y tiene la data de tres  
-- departamentos del país, por lo cual les pide su  
-- opinión sobre qué productos sobresalen, que  
-- combinaciones son mejores y quieren este estudio  
-- por departamento y también por país.  
-- Materia: DATAWAREHOUSE Y MINERÍA DE DATOS  
-- Archivo: Analisis_Fiorella.sql  
-- Descripción: Ingreso de datos ejemplo para base de datos  
-- Alumnos: Driotis Cruz, David Otoniel.....DC211280  
-- Flores Quintanilla, Roberto  
-- Carlos.....FQ211776  
-- Código fuente: https://github.com/DMD941/Desafio\_01  
-- -----
```

Con este script hacemos un DROP de la base de datos si ya existía anteriormente de una forma correcta, cerrando cualquier conexión a ella que este activa.

```
--Remove backup history  
EXEC msdb.dbo.sp_delete_database_backuphistory @database_name =  
N'EJERCICIO2'  
GO  
USE [master]  
GO  
--Cerramos todas las conexiones a la base de datos  
ALTER DATABASE [EJERCICIO1] SET SINGLE_USER WITH ROLLBACK IMMEDIATE  
GO  
USE [master]  
GO  
--removemos la base de datos, si esta existe previamente  
DROP DATABASE IF EXISTS [EJERCICIO2]  
GO  
-- -----
```

```
--Creamos la base de datos EJERCICIO2
USE master;
CREATE DATABASE [EJERCICIO2]
GO
-- -----
```

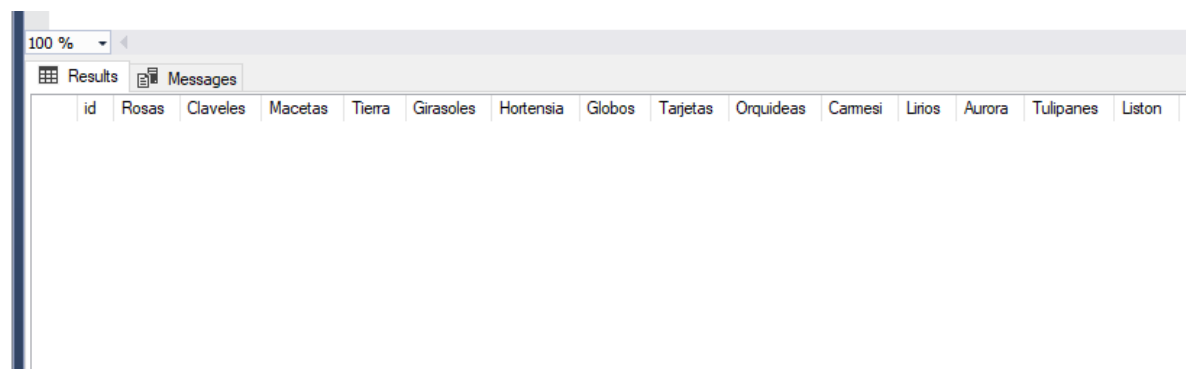
```
--Removemos la tabla Floristeria_San_Miguel, si existiera previamente
DROP TABLE IF EXISTS [EJERCICIO2].[dbo].[Floristeria_San_Miguel]
GO
--Creamos la tabla Floristeria_San_Miguel
CREATE TABLE [EJERCICIO2].[dbo].[Floristeria_San_Miguel] (
    [id] varchar(50),
    [Rosas] INT,
    [Claveles] INT,
    [Macetas] INT,
    [Tierra] INT,
    [Girasoles] INT,
    [Hortensia] INT,
    [Globos] INT,
    [Tarjetas] INT,
    [Orquideas] INT,
    [Carmesi] INT,
    [Lirios] INT,
    [Aurora] INT,
    [Tulipanes] INT,
    [Liston] INT
)
GO
--Verificamos que la tabla fue creada al mostrar los primeros 1000
registros
SELECT TOP (1000) [id]
    , [Rosas]
    , [Claveles]
    , [Macetas]
    , [Tierra]
    , [Girasoles]
    , [Hortensia]
    , [Globos]
    , [Tarjetas]
    , [Orquideas]
    , [Carmesi]
    , [Lirios]
    , [Aurora]
    , [Tulipanes]
    , [Liston]
FROM [EJERCICIO2].[dbo].[Floristeria_San_Miguel]
-----
```

100 %														
Results	Messages													
id	Rosas	Claveles	Macetas	Tierra	Girasoles	Hortensia	Globos	Tarjetas	Orquideas	Carnesi	Lirios	Aurora	Tulpanes	Liston

San Salvador

```
--Removemos la tabla Floristeria_San_Salvador, si existiera previamente
DROP TABLE IF EXISTS [EJERCICIO2].[dbo].[Floristeria_San_Salvador]
GO
--Creamos la tabla Floristeria_San_Salvador
CREATE TABLE [EJERCICIO2].[dbo].[Floristeria_San_Salvador] (
    [id] varchar(50),
    [Rosas] INT,
    [Claveles] INT,
    [Macetas] INT,
    [Tierra] INT,
    [Girasoles] INT,
    [Hortensia] INT,
    [Globos] INT,
    [Tarjetas] INT,
    [Orquideas] INT,
    [Carmesi] INT,
    [Lirios] INT,
    [Aurora] INT,
    [Tulipanes] INT,
    [Liston] INT
)
GO
--Verificamos que la tabla fue creada al mostrar los primeros 1000
registros
SELECT TOP (1000) [id]
    , [Rosas]
    , [Claveles]
    , [Macetas]
    , [Tierra]
    , [Girasoles]
    , [Hortensia]
    , [Globos]
    , [Tarjetas]
    , [Orquideas]
    , [Carmesi]
    , [Lirios]
    , [Aurora]
    , [Tulipanes]
    , [Liston]
FROM [EJERCICIO2].[dbo].[Floristeria_San_Salvador]
-----
```

Verificamos que la tabla Floristeria_San_Salvador fue creada al mostrar los primeros 1000 registros



id	Rosas	Claveles	Macetas	Tierra	Girasoles	Hortensia	Globos	Tarjetas	Orquideas	Carmesi	Lirios	Aurora	Tulipanes	Liston
----	-------	----------	---------	--------	-----------	-----------	--------	----------	-----------	---------	--------	--------	-----------	--------

Santa Ana

```
--Removemos la tabla Floristeria_Santa_Ana, si existiera previamente
DROP TABLE IF EXISTS [EJERCICIO2].[dbo].[Floristeria_Santa_Ana]
GO
--Creamos la tabla Floristeria_Santa_Ana
CREATE TABLE [EJERCICIO2].[dbo].[Floristeria_Santa_Ana] (
    [id] varchar(50),
    [Rosas] INT,
    [Claveles] INT,
    [Macetas] INT,
    [Tierra] INT,
    [Girasoles] INT,
    [Hortensia] INT,
    [Globos] INT,
    [Tarjetas] INT,
    [Orquideas] INT,
    [Carmesi] INT,
    [Lirios] INT,
    [Aurora] INT,
    [Tulipanes] INT,
    [Liston] INT
)
GO
--Verificamos que la tabla fue creada al mostrar los primeros 1000
registros
SELECT TOP (1000) [id]
    , [Rosas]
    , [Claveles]
    , [Macetas]
    , [Tierra]
    , [Girasoles]
    , [Hortensia]
    , [Globos]
    , [Tarjetas]
    , [Orquideas]
    , [Carmesi]
    , [Lirios]
    , [Aurora]
    , [Tulipanes]
    , [Liston]
FROM [EJERCICIO2].[dbo].[Floristeria_Santa_Ana]
```

Verificamos que la tabla Floristeria_Santa_Ana fue creada al mostrar los primeros 1000 registros

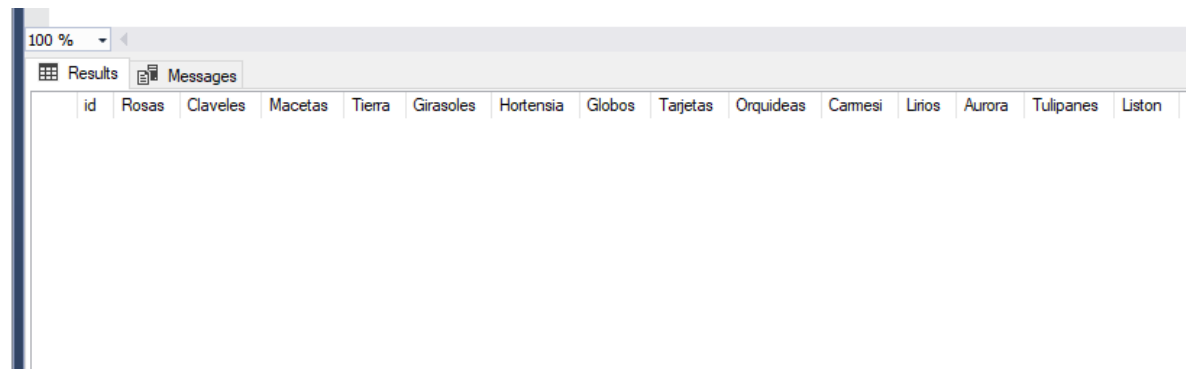


id	Rosas	Claveles	Macetas	Tierra	Girasoles	Hortensia	Globos	Tarjetas	Orquideas	Carmesi	Lirios	Aurora	Tulipanes	Liston
----	-------	----------	---------	--------	-----------	-----------	--------	----------	-----------	---------	--------	--------	-----------	--------

Floristeria pais

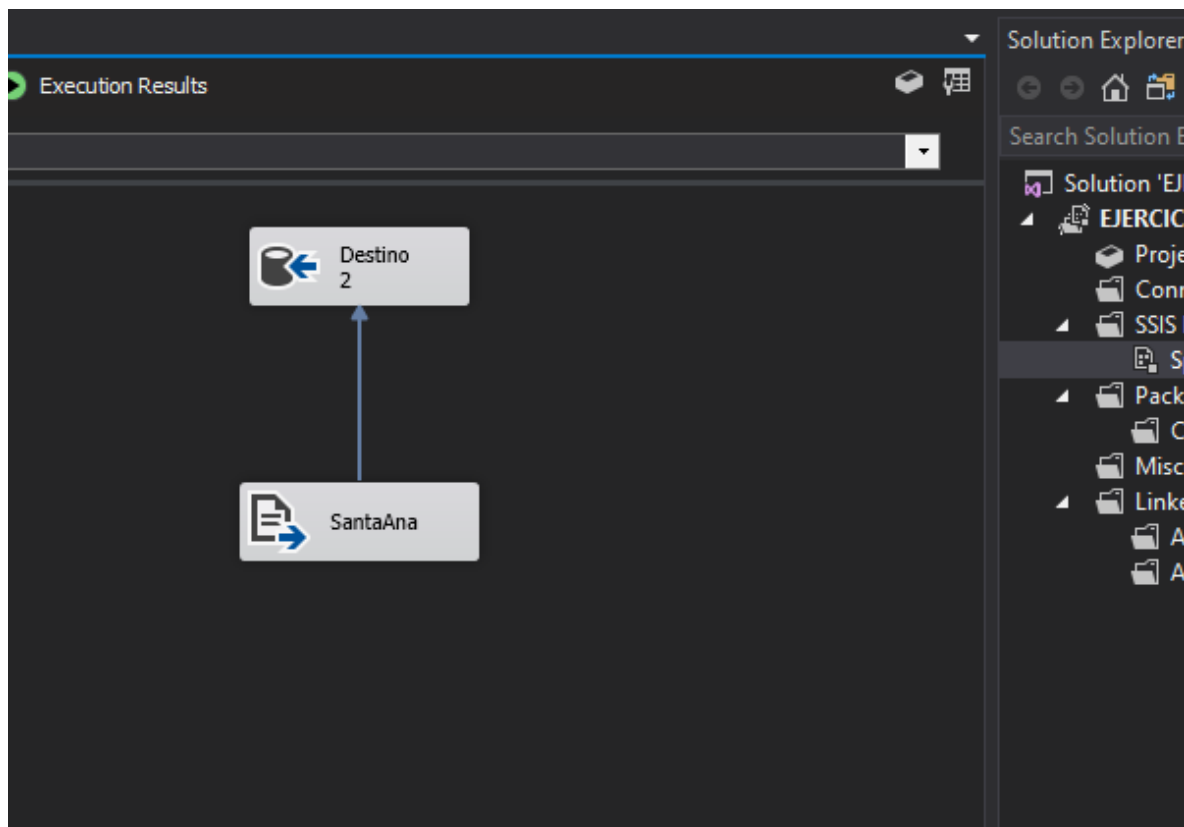
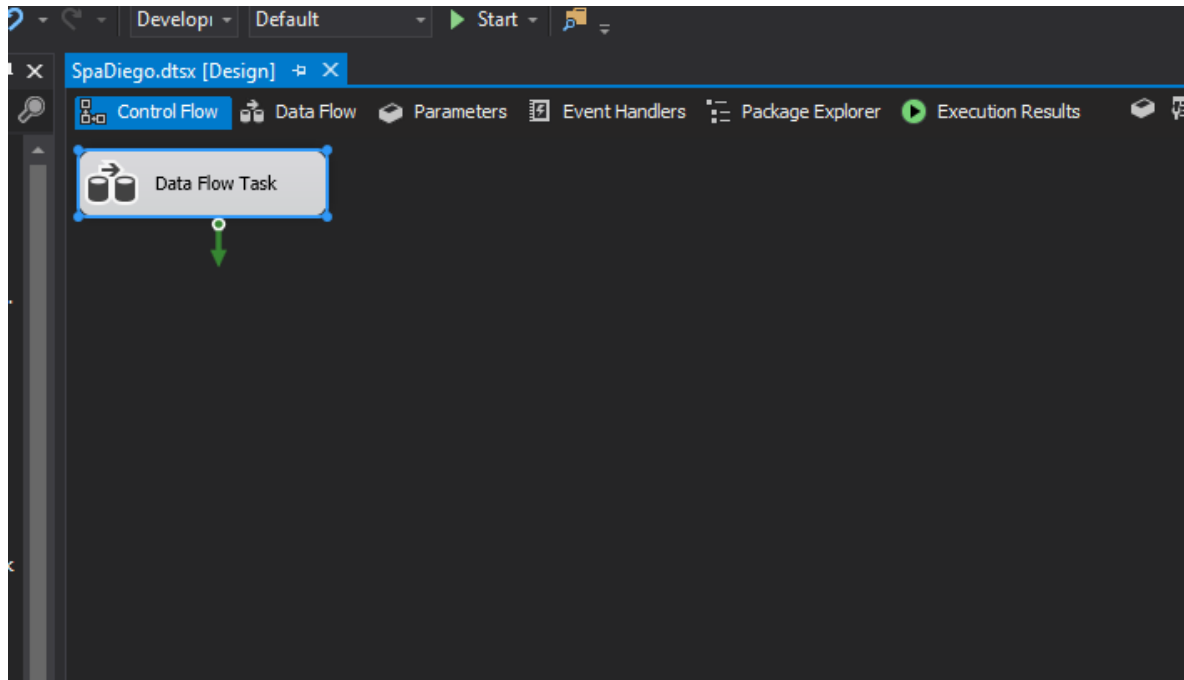
```
--Removemos la tabla Floristeria_Pais, si existiera previamente
DROP TABLE IF EXISTS [EJERCICIO2].[dbo].[Floristeria_Pais]
GO
--Creamos la tabla Floristeria_Pais
CREATE TABLE [EJERCICIO2].[dbo].[Floristeria_Pais] (
    [id] varchar(50),
    [Rosas] INT,
    [Claveles] INT,
    [Macetas] INT,
    [Tierra] INT,
    [Girasoles] INT,
    [Hortensia] INT,
    [Globos] INT,
    [Tarjetas] INT,
    [Orquideas] INT,
    [Carmesi] INT,
    [Lirios] INT,
    [Aurora] INT,
    [Tulipanes] INT,
    [Liston] INT
)
GO
--Verificamos que la tabla fue creada al mostrar los primeros 1000
registros
SELECT TOP (1000) [id]
    , [Rosas]
    , [Claveles]
    , [Macetas]
    , [Tierra]
    , [Girasoles]
    , [Hortensia]
    , [Globos]
    , [Tarjetas]
    , [Orquideas]
    , [Carmesi]
    , [Lirios]
    , [Aurora]
    , [Tulipanes]
    , [Liston]
FROM [EJERCICIO2].[dbo].[Floristeria_Pais]
```

Verificamos que las tabla Floristeria_Pais fue creada al mostrar los primeros 1000 registros

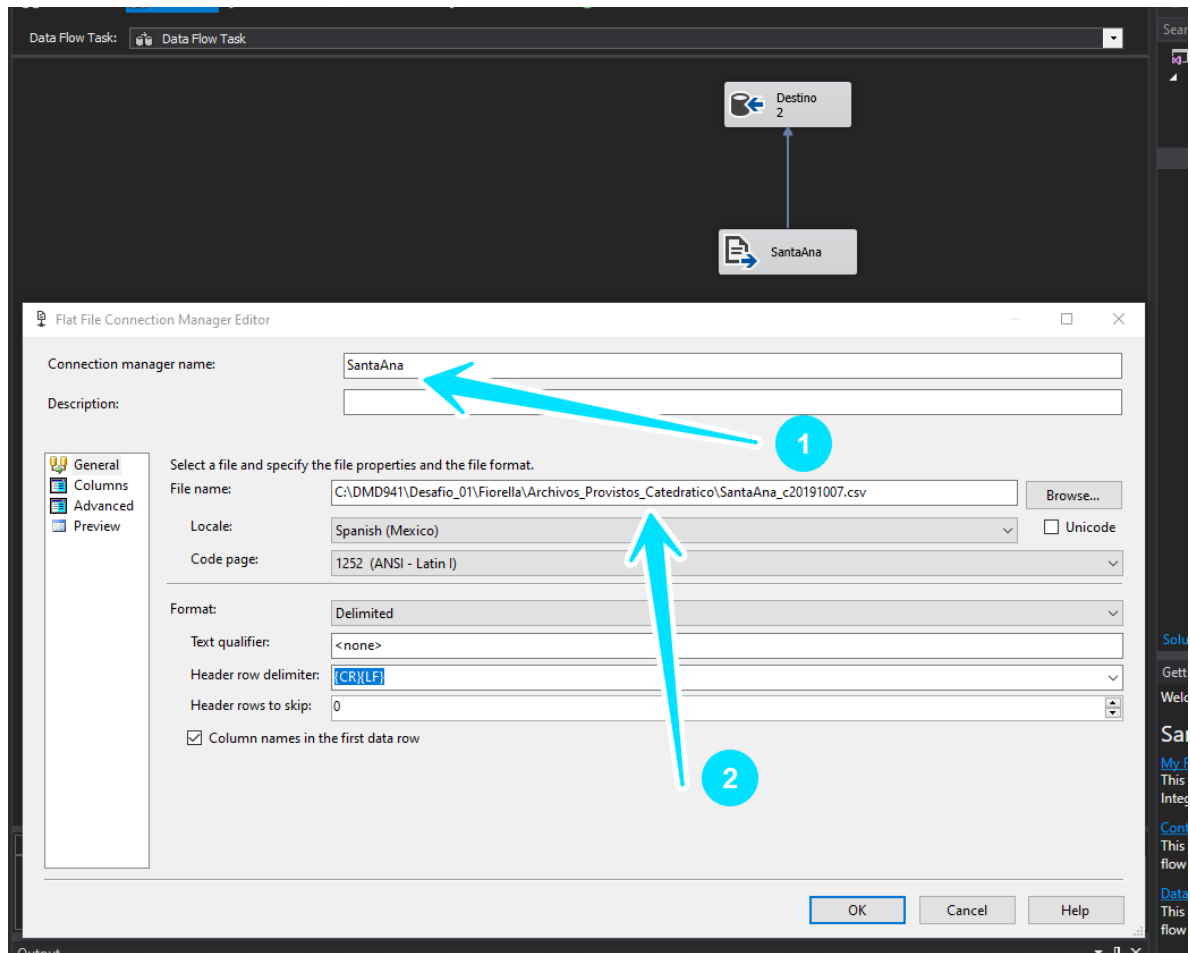


id	Rosas	Claveles	Macetas	Tierra	Girasoles	Hortensia	Globos	Tarjetas	Orquideas	Carmesi	Lirios	Aurora	Tulipanes	Liston
----	-------	----------	---------	--------	-----------	-----------	--------	----------	-----------	---------	--------	--------	-----------	--------

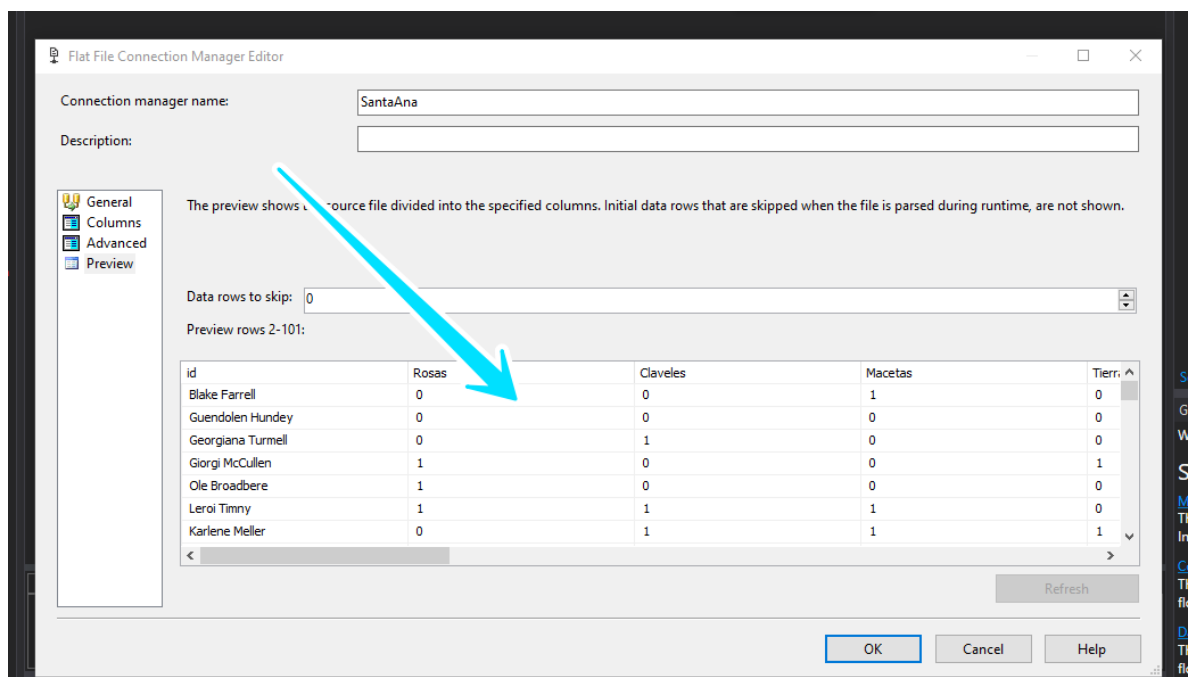
Procedemos a crear en SQL Server integration services dentro de nuestro Visual Studio nuestro Flow de datos



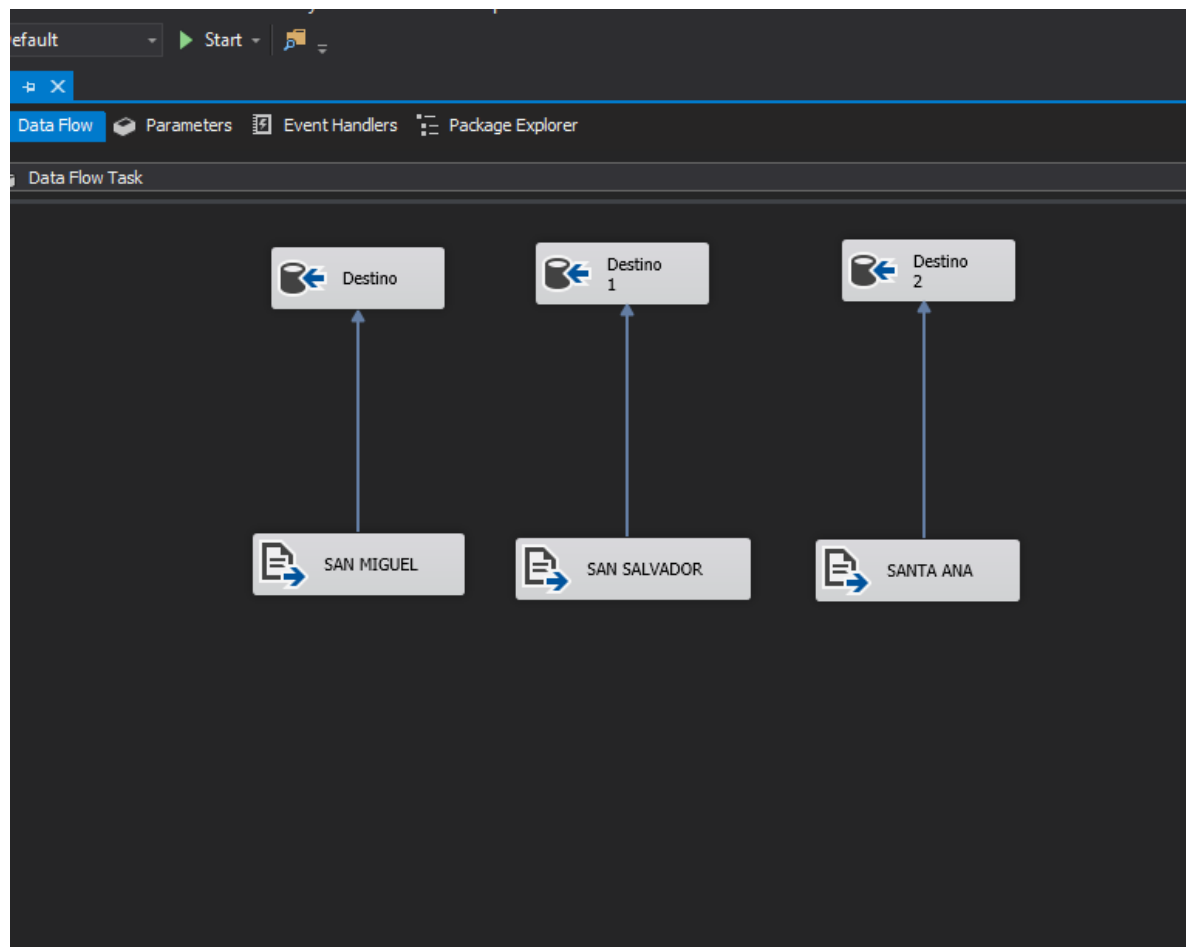
1. Procedemos a nombrar nuestra conexión a nuestro archivo plano
2. Marcamos la ruta a nuestro archivo plano de origen



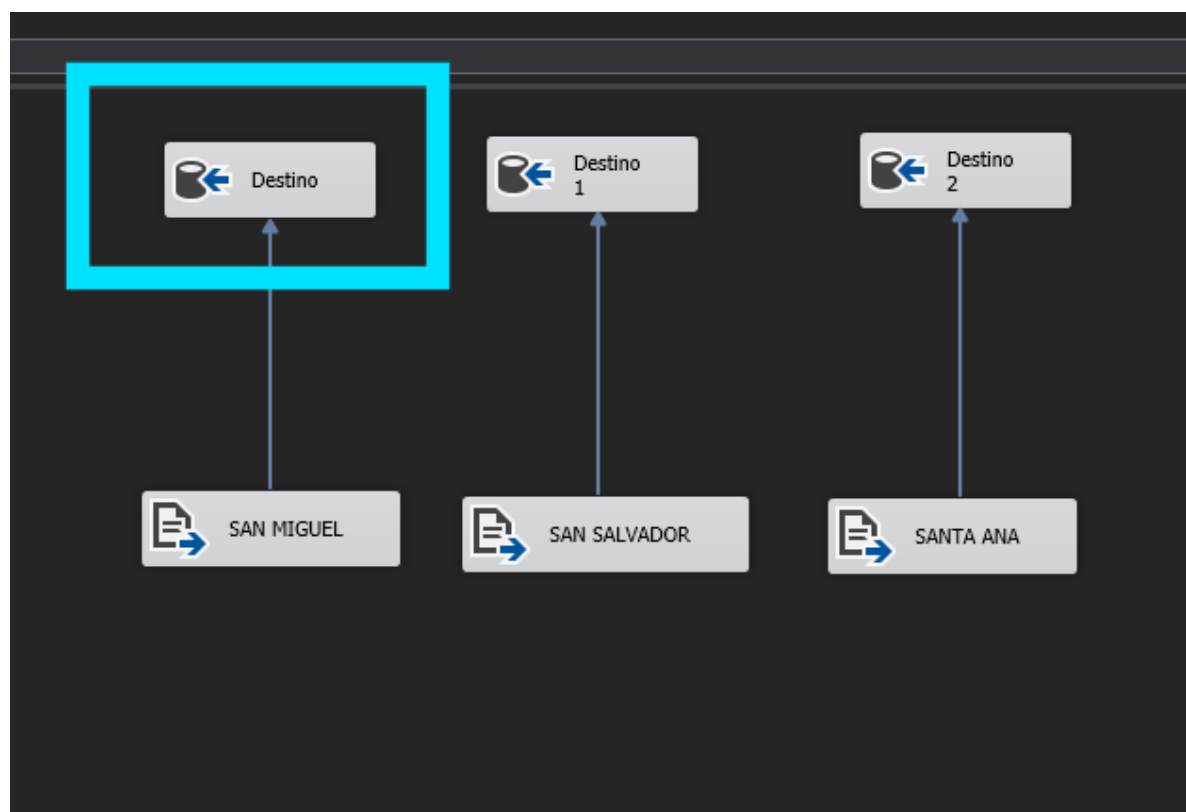
En esta parte podemos ver una vista previa de los datos.



Hacemos eso por cada una de las sucursales.



En nuestro OLE DB destination definimos la tabla donde se vaciaran los datos



Configure the properties used to insert data into a relational database using an OLE DB provider.

Configure OLE DB Connection Manager

To create a connection manager based on previously defined connection information, select a data connection, and then click OK. To create a new connection manager, click New.

Data connections:

DESKTOP-VD039RD.EJERCICIO2

Data connection properties:

Property	Value
Data Source	DESKTOP-VD039RD
Initial Catalog	EJERCICIO2
Integrated Se...	SSPI
Provider	SQLNCLI11.1

New...

Delete

OK

Cancel

Configure the properties used to insert data into a relational database using an OLE DB provider.

Connection Manager
Mappings
Error Output

Specify an OLE DB connection manager, a data source, or a data source view, and select the data access mode. If using the SQL command access mode, specify the SQL command either by typing the query or by using Query Builder. For fast-load data access, set the table update options.

OLE DB connection manager:

DESKTOP-VD039RD.EJERCICIO2

New...

Data access mode:

Table or view - fast load

Name of the table or the view:

[dbo].[Floristeria_San_Miguel]

New...

☐ Keep identity

☒ Table lock

☐ Keep nulls

☒ Check constraints

Rows per batch:

Maximum insert commit size:

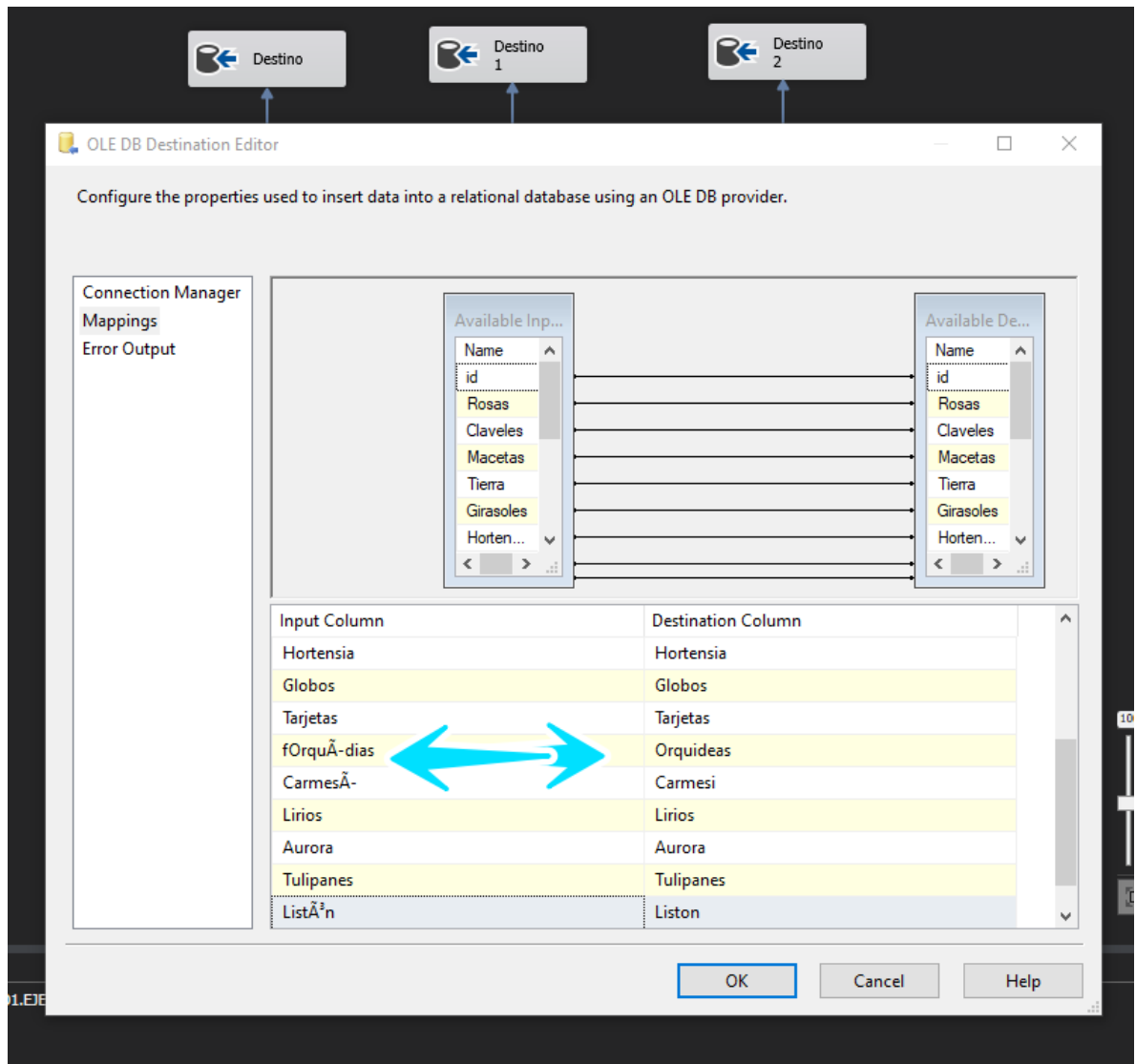
2147483647

View Existing

OK

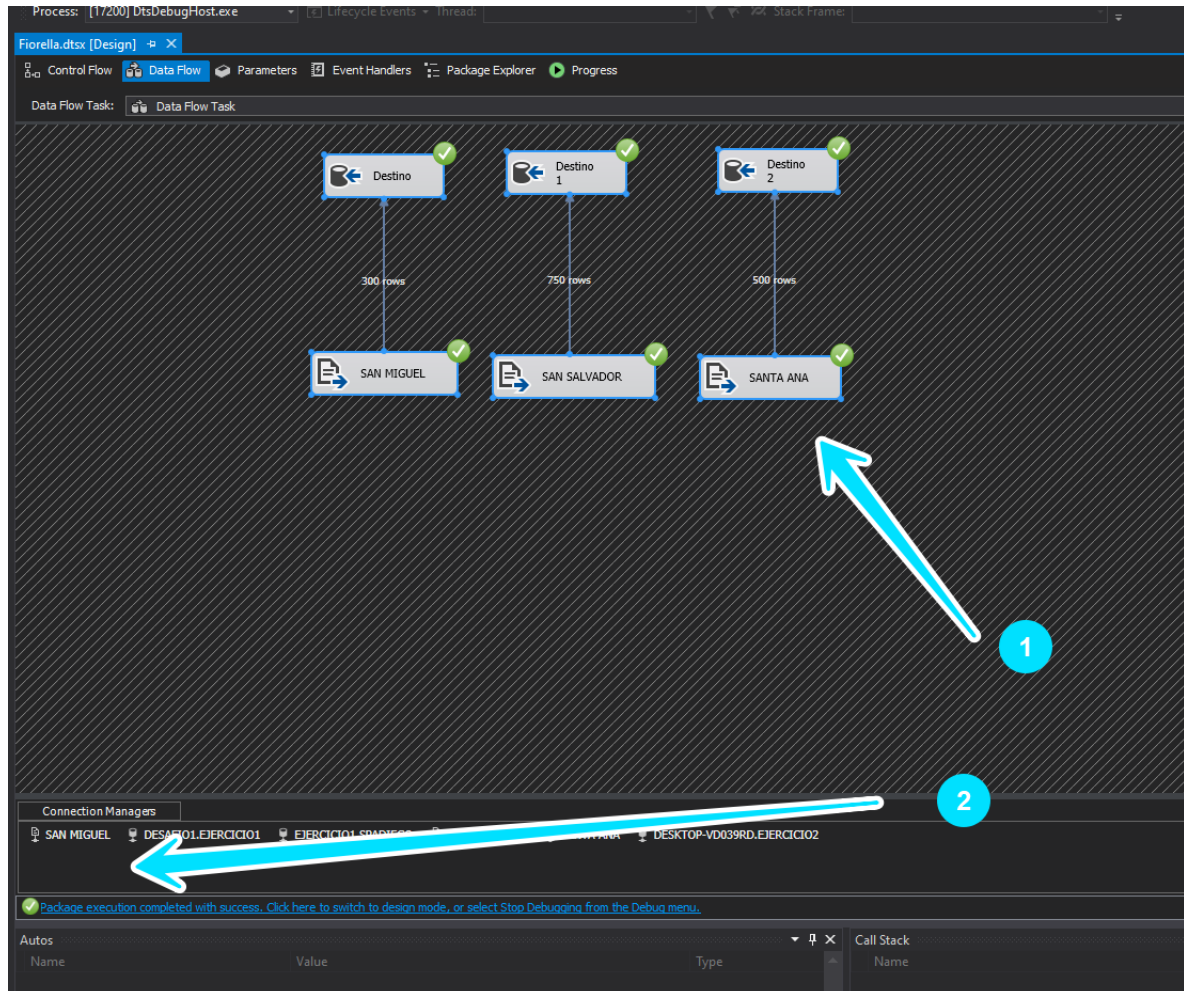
Cancel

Help



Procedemos a:

1. Ejecutar nuestro proyecto
2. Corroborar que se ha efectuado correctamente



Ahora en **SQL Server Management Studio** Verificamos que la tabla fue actualizada al mostrar los primeros 1000 registros

```
--Verificamos que la tabla fue creada al mostrar los primeros 1000 registros
SELECT TOP (1000) [id]
      ,[Rosas]
      ,[Claveles]
      ,[Macetas]
      ,[Tierra]
      ,[Girasoles]
      ,[Hortensia]
      ,[Globos]
      ,[Tarjetas]
      ,[Orquideas]
      ,[Carmesi]
      ,[Lirios]
      ,[Aurora]
      ,[Tulipanes]
      ,[Liston]
FROM [EJERCICIO2].[dbo].[Floristeria_San_Salvador]
```

SQLQuery19.sql - D...9RD\Roberto (139)) SQLQuery18.sql - D...39RD\Roberto (91)) SQLQuery17.sql - D...9RD\Roberto (91))

```

/***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP (1000) [id]
, [Rosas]
, [Claveles]
, [Macetas]
, [Tierra]
, [Girasoles]
, [Hortensia]
, [Globos]
, [Tarjetas]
, [Orquideas]
, [Carmesi]
, [Lirios]
, [Aurora]
, [Tulipanes]
, [Liston]
FROM [EJERCICIO2].[dbo].[Floristeria_San_Salvador]

```

100 %

Results Messages

	id	Rosas	Claveles	Macetas	Tierra	Girasoles	Hortensia	Globos	Tarjetas	Orquideas	Carmesi
1	Loren Pritty	1	0	1	1	0	0	1	1	1	1
2	Curran Lackey	1	1	1	1	0	0	0	1	1	1
3	Marietta Luff	1	1	0	1	0	0	1	1	1	1
4	Codi Finnemore	1	0	1	0	0	1	0	1	1	1
5	Keane McMains	1	0	0	1	0	0	0	0	1	0
6	Markus Fursland	1	1	0	0	0	1	1	0	0	1
7	Davide Paulou	0	0	0	1	1	1	0	1	1	0
8	Valentine Giottoi	1	0	1	1	0	0	0	0	0	0
9	Atalanta O'Hagirtie	1	0	0	1	0	0	1	1	1	1
10	Roze Cratchley	1	1	0	1	1	1	1	1	1	0
11	Milicent Bewsey	1	0	1	0	1	1	1	1	0	0
12	Missy Galbreth	1	1	1	1	1	0	1	1	0	0
13	Opal Stopper	1	1	1	1	1	0	1	1	0	1
14	Dot Joslin	0	1	0	0	0	1	1	1	1	0
15	Darbee Sylvester	1	0	1	0	0	0	1	1	1	0
16	Roseann Kinning	1	0	1	1	1	0	1	0	1	0

Ahora procedemos a interpretar nuestros datos por departamento:

Ejemplo para San Miguel

```
-- Script para conocer las ventas de cada producto de San Miguel
SELECT
    sum(Rosas) as Rosas
    ,sum(Claveles) as Claveles
    ,sum(Macetas) as Macetas
    ,sum(Tierra) as Tierra
    ,sum(Girasoles) as Girasoles
    ,sum(Hortensia) as Hortensia
    ,sum(Globos) as Globos
    ,sum(Tarjetas) as Tarjetas
    ,sum(Orquideas) as Orquideas
    ,sum(Carmesi) as Carmesi
    ,sum(Lirios) as Lirios
    ,sum(Aurora) as Aurora
    ,sum(Tulipanes) as Tulipanes
    ,sum(Liston) as Liston
FROM [EJERCICIO2].[dbo].[Floristeria_San_Miguel]
```

SQLQuery6.sql - DE...9RD\Roberto (141))* X SQLQuery5.sql - DE...9RD\Roberto (136)) SQLQuery4.sql - DE...9RD\Roberto (133))

```
-- Script para conocer las ventas de cada producto de San Miguel
SELECT
    sum(Rosas) as Rosas
    ,sum(Claveles) as Claveles
    ,sum(Macetas) as Macetas
    ,sum(Tierra) as Tierra
    ,sum(Girasoles) as Girasoles
    ,sum(Hortensia) as Hortensia
    ,sum(Globos) as Globos
    ,sum(Tarjetas) as Tarjetas
    ,sum(Orquideas) as Orquideas
    ,sum(Carmesi) as Carmesi
    ,sum(Lirios) as Lirios
    ,sum(Aurora) as Aurora
    ,sum(Tulipanes) as Tulipanes
    ,sum(Liston) as Liston
FROM [EJERCICIO2].[dbo].[Floristeria_San_Miguel]
```

100 %

Results Messages

	Rosas	Claveles	Macetas	Tierra	Girasoles	Hortensia	Globos	Tarjetas	Orquideas	Carmesi	Lirios	Aurora	Tulipanes	Liston
1	157	137	141	141	150	157	151	143	158	158	160	160	149	149

menos vendidos

mas vendidos

Concluimos que para San Miguel los productos más vendidos son: “Los lirios” y “Aurora”, siendo los menos vendidos las “Macetas” y “Tierra”

Sucursal San Salvador:

SQLQuery6.sql - DE...9RD\Roberto (141)) * SQLQuery5.sql - DE...9RD\Roberto (136)) SQLQuery4.sql - DE...9RD\Roberto (133))

```
-- Script para conocer las ventas de cada producto de San Salvador
SELECT
    sum(Rosas) as Rosas
    ,sum(Claveles) as Claveles
    ,sum(Macetas) as Macetas
    ,sum(Tierra) as Tierra
    ,sum(Girasoles) as Girasoles
    ,sum(Hortensia) as Hortensia
    ,sum(Globos) as Globos
    ,sum(Tarjetas) as Tarjetas
    ,sum(Orquideas) as Orquideas
    ,sum(Carmesi) as Carmesi
    ,sum(Lirios) as Lirios
    ,sum(Aurora) as Aurora
    ,sum(Tulipanes) as Tulipanes
    ,sum(Liston) as Liston
FROM [EJERCICIO2].[dbo].[Floristeria_San_Salvador]
```

100 %

Results Messages

	Rosas	Claveles	Macetas	Tierra	Girasoles	Hortensia	Globos	Tarjetas	Orquideas	Carmesi	Lirios	Aurora	Tulipanes	Liston
1	612	350	392	368	371	374	587	384	380	353	365	384	357	690

Concluimos que para San Salvador los productos más vendidos son: “Liston”
“Rosas” y “Globos”, siendo los menos vendidos las “Claveles” y “Carmesi”

Sucursal Santa Ana:

SQLQuery6.sql - DE...9RD\Roberto (141)) * SQLQuery5.sql - DE...9RD\Roberto (136)) SQLQuery4.sql - DE...9RD\Roberto (133))

```
-- Script para conocer las ventas de cada producto de Santa Ana
SELECT
    sum(Rosas) as Rosas
    ,sum(Claveles) as Claveles
    ,sum(Macetas) as Macetas
    ,sum(Tierra) as Tierra
    ,sum(Girasoles) as Girasoles
    ,sum(Hortensia) as Hortensia
    ,sum(Globos) as Globos
    ,sum(Tarjetas) as Tarjetas
    ,sum(Orquideas) as Orquideas
    ,sum(Carmesi) as Carmesi
    ,sum(Lirios) as Lirios
    ,sum(Aurora) as Aurora
    ,sum(Tulipanes) as Tulipanes
    ,sum(Liston) as Liston
FROM [EJERCICIO2].[dbo].[Floristeria_Santa_Ana]
```

100 %

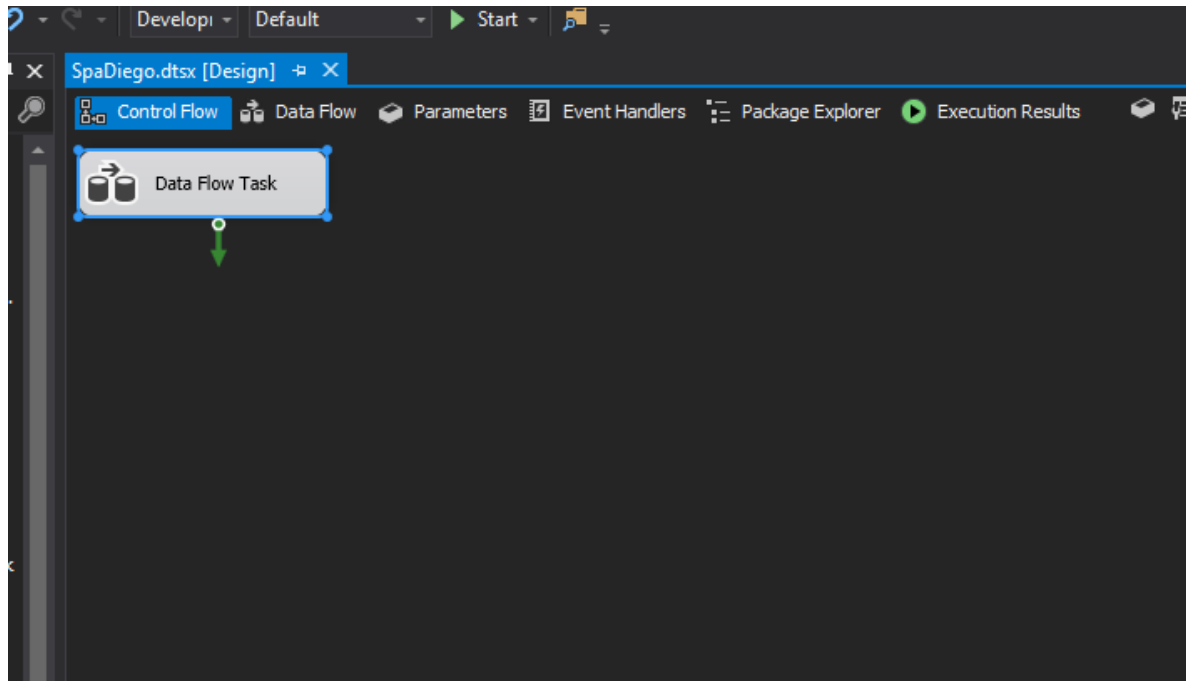
Results Messages

	Rosas	Claveles	Macetas	Tierra	Girasoles	Hortensia	Globos	Tarjetas	Orquideas	Carmesi	Lirios	Aurora	Tulipanes	Liston
1	176	246	245	236	266	243	154	252	259	236	270	260	247	136

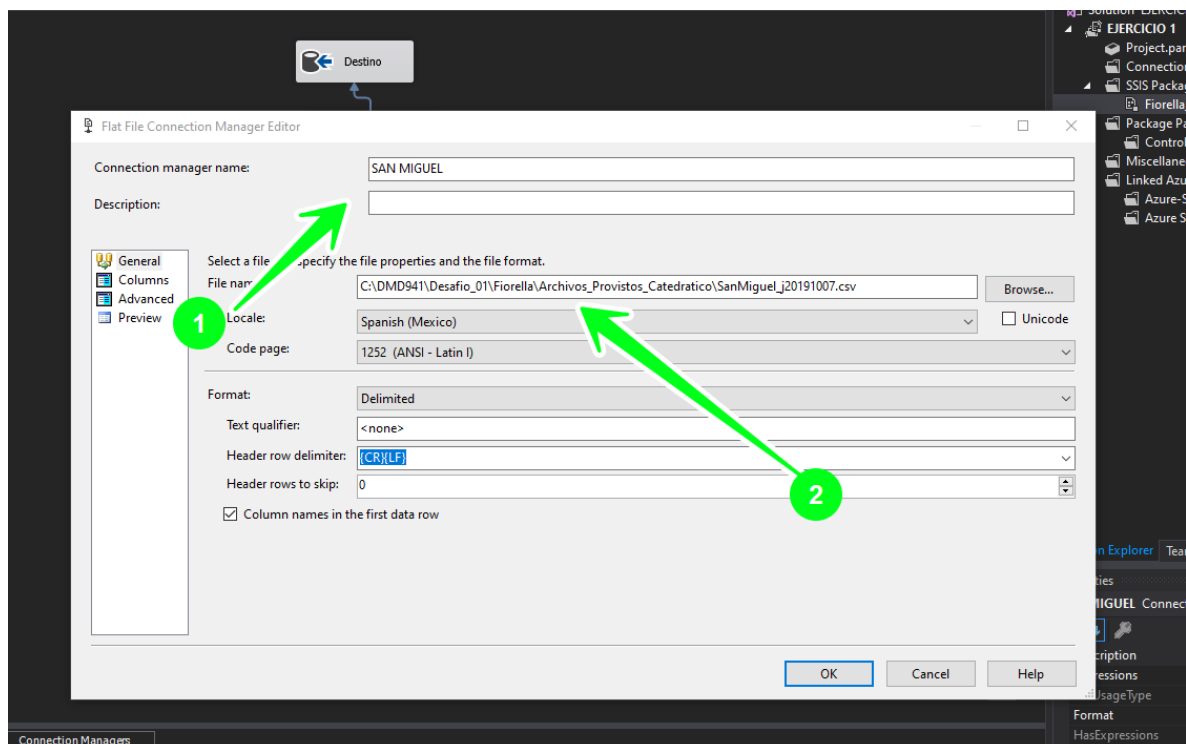
Respecto a la sucursal Santa Ana concluimos que lo mas vendidos son los
“Lirios” y “Aurora” y lo menos demandado es “Globos” y “listón”

Ahora procedemos al análisis a nivel de país:

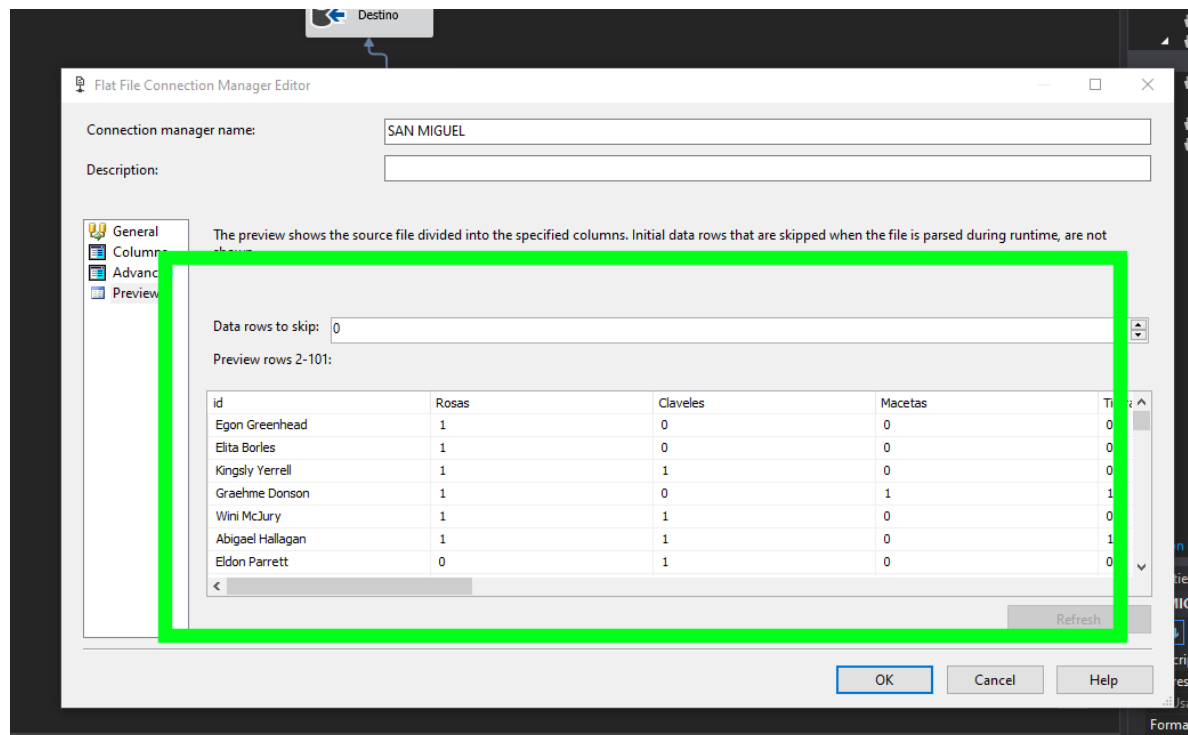
Procedemos a crear en SQL Server integration services dentro de nuestro Visual Studio nuestro Flow de datos



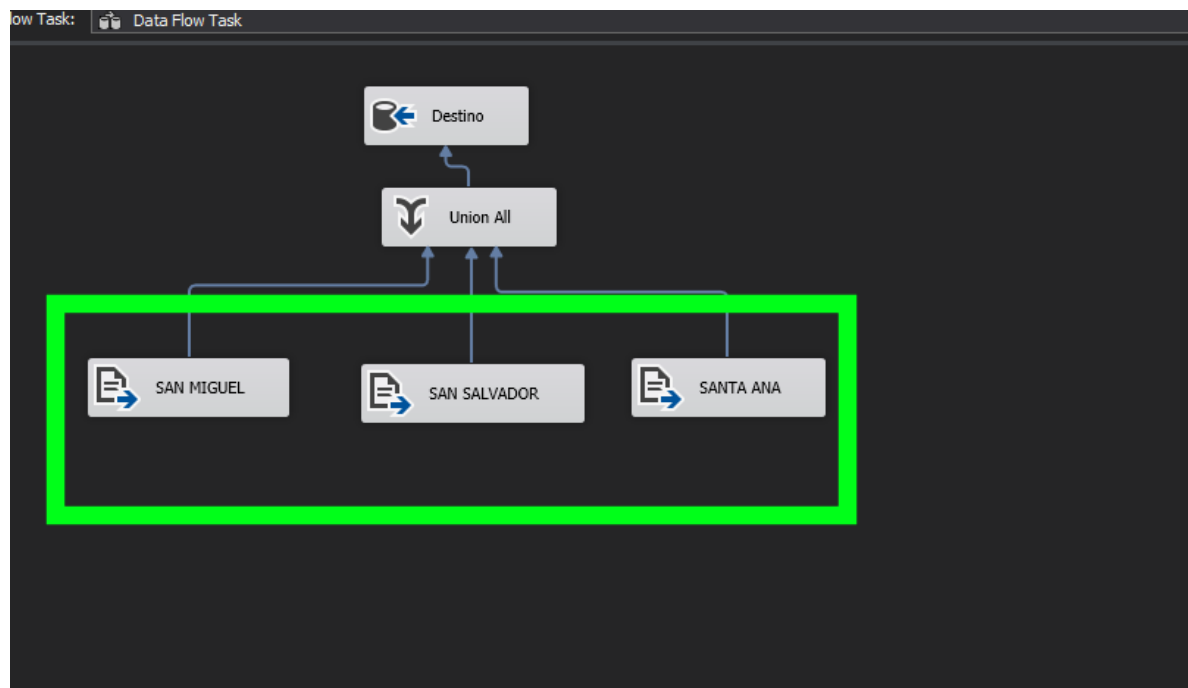
1. Procedemos a nombrar nuestra conexión a nuestro archivo plano
2. Marcamos la ruta a nuestro archivo plano de origen



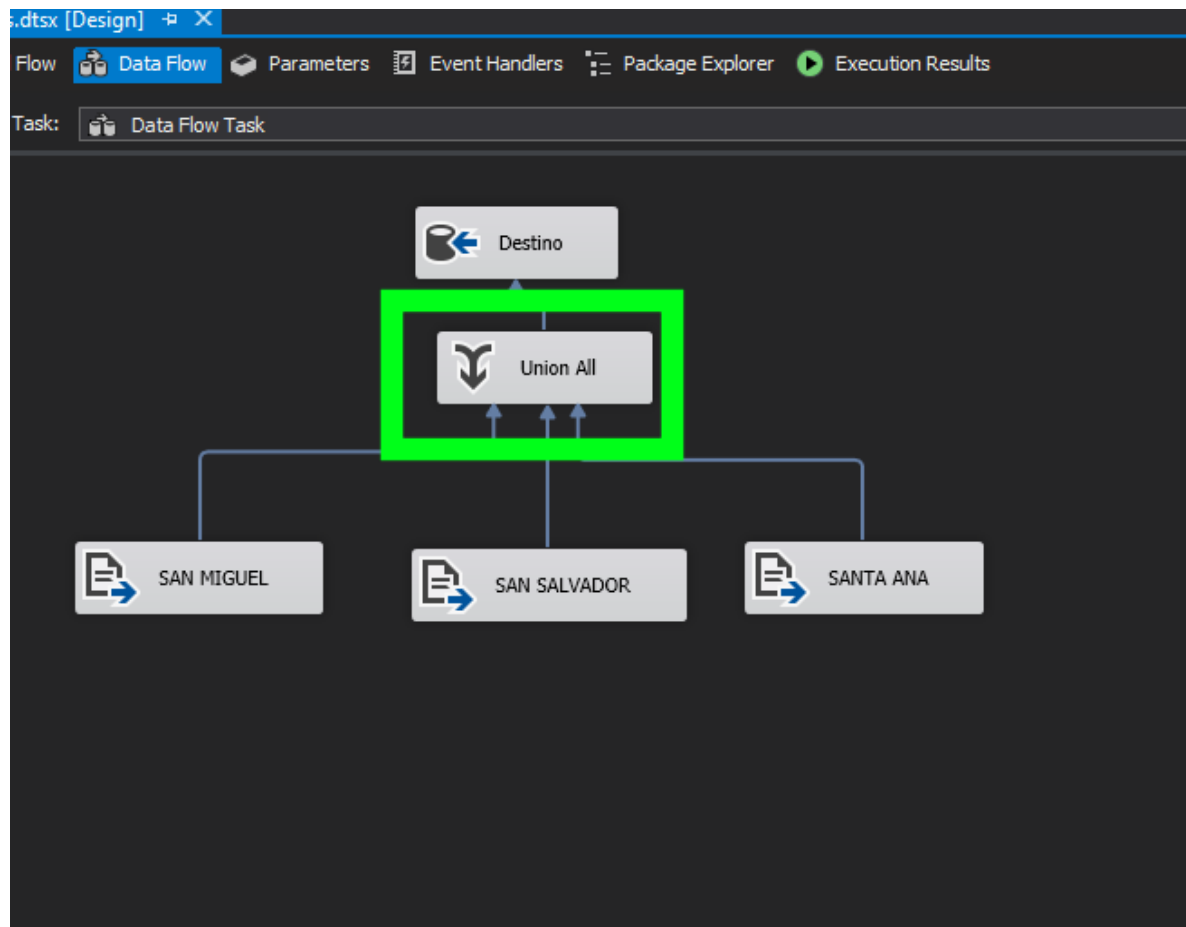
En esta parte podemos ver una vista previa de los datos.



Hacemos eso por cada una de las sucursales.



Configuramos como se llevara a cabo la unión de los registros



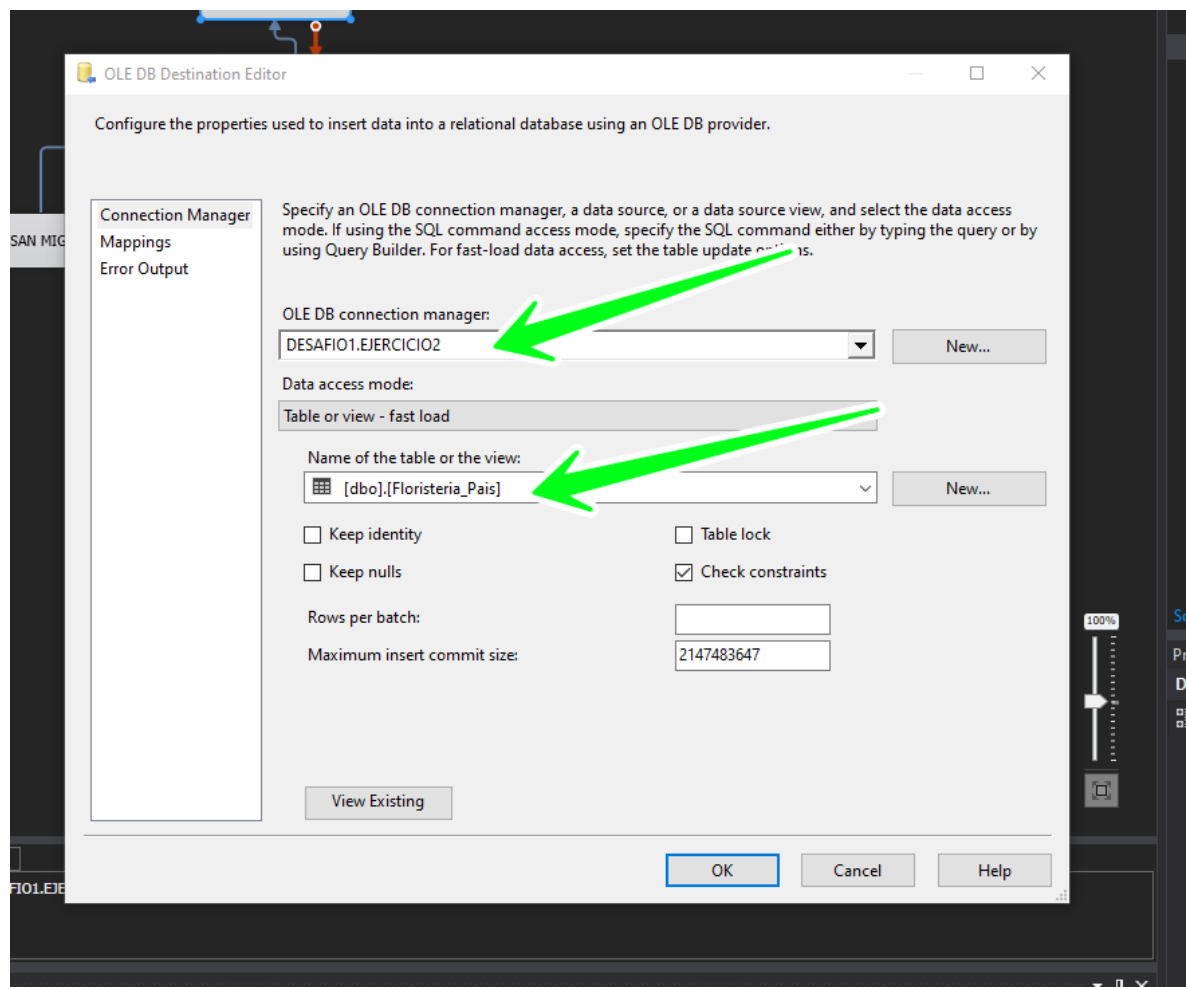
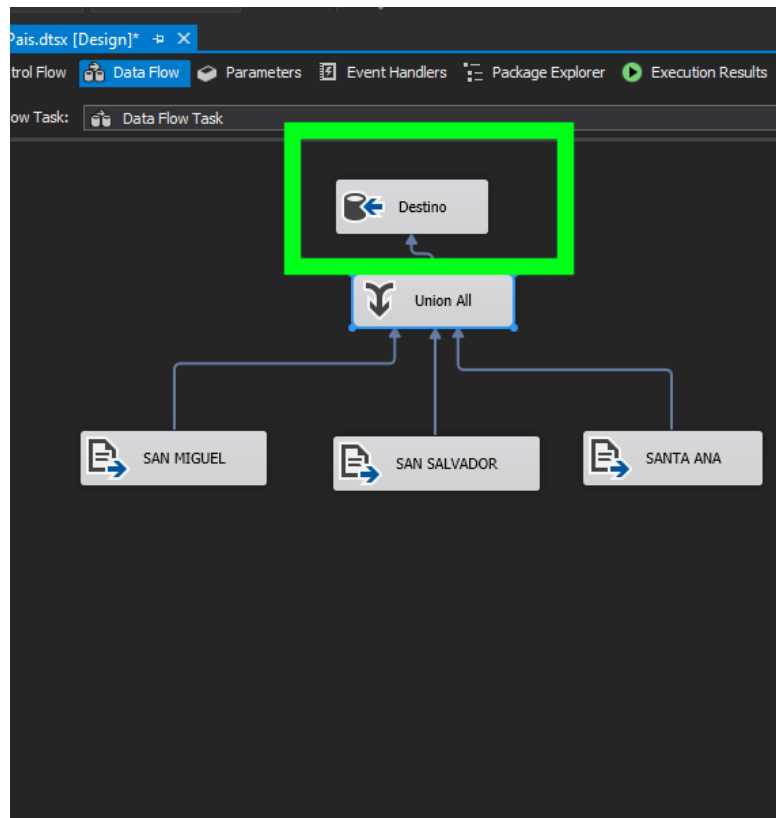
Union All Transformation Editor

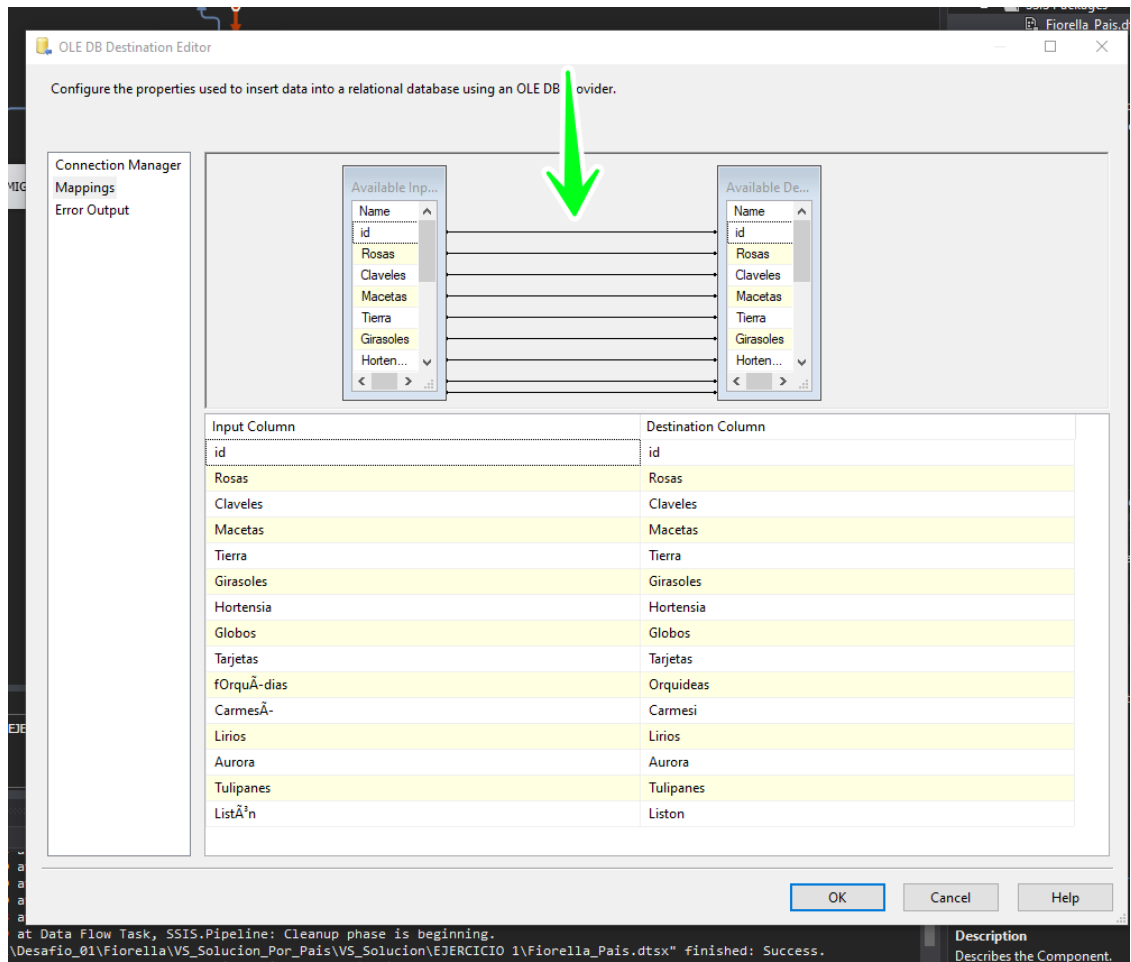
Configure the properties used to merge multiple inputs into one output by creating mappings between columns.

Output Column Name	Union All Input 1	Union All Input 2	Union All Input 3
id	id	id	id
Rosas	Rosas	Rosas	Rosas
Claveles	Claveles	Claveles	Claveles
Macetas	Macetas	Macetas	Macetas
Tierra	Tierra	Tierra	Tierra
Girasoles	Girasoles	Girasoles	Girasoles
Hortensia	Hortensia	Hortensia	Hortensia
Globos	Globos	Globos	Globos
Tarjetas	Tarjetas	Tarjetas	Tarjetas
fOrquÃ-dias	fOrquÃ-dias	fOrquÃ-dias	fOrquÃ-dias
CarmesÃ-	CarmesÃ-	CarmesÃ-	CarmesÃ-
Lirios	Lirios	Lirios	Lirios
Aurora	Aurora	Aurora	Aurora
Tulipanes	Tulipanes	Tulipanes	Tulipanes
ListÃn	ListÃn	ListÃn	ListÃn

OK Cancel Help

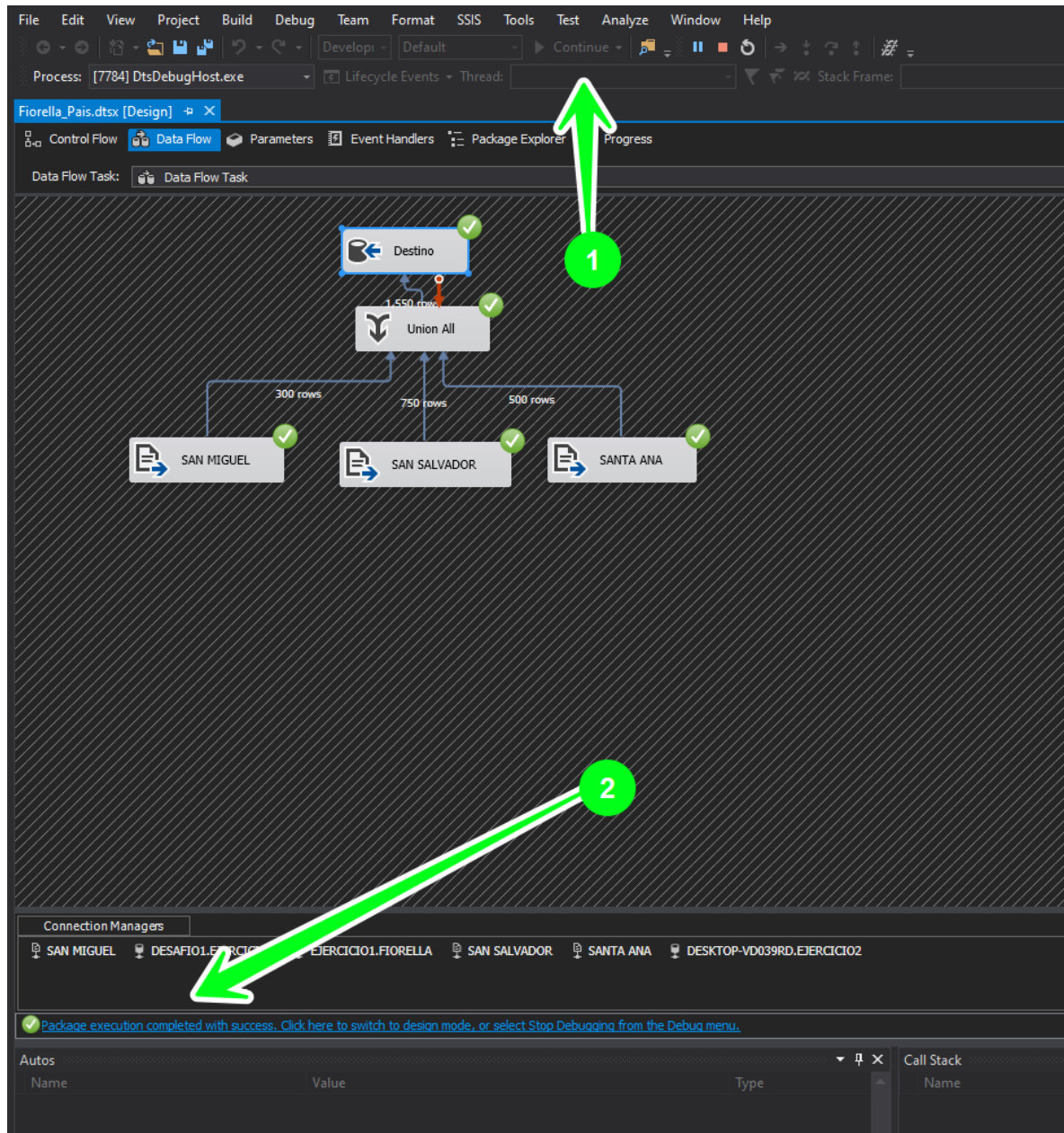
En nuestro OLE DB destination definimos la tabla donde se vaciaran los datos





Procedemos a:

1. Ejecutar nuestro proyecto
2. Corroborar que se ha efectuado correctamente



```
--Verificamos que la tabla fue creada al mostrar los primeros 1000 registros
SELECT TOP (1000) [id]
      ,[Rosas]
      ,[Claveles]
      ,[Macetas]
      ,[Tierra]
      ,[Girasoles]
      ,[Hortensia]
      ,[Globos]
      ,[Tarjetas]
      ,[Orquideas]
      ,[Carmesi]
      ,[Lirios]
      ,[Aurora]
      ,[Tulipanes]
      ,[Liston]
FROM [EJERCICIO2].[dbo].[Floristeria_Pais]
```

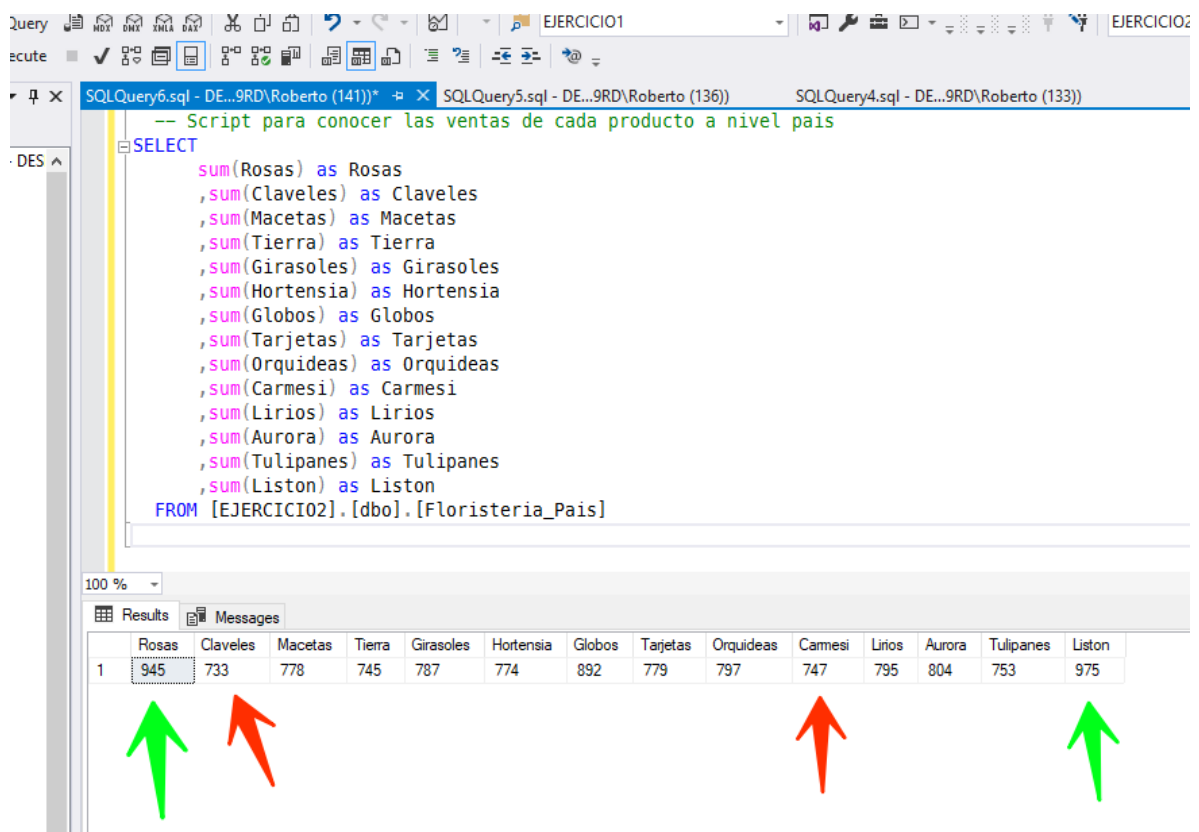
[illegible]

Ahora procedemos a interpretar nuestros datos como pais

```
-- Script para conocer las ventas de cada producto a nivel pais  
SELECT
```

```
    sum(Rosas) as Rosas  
    ,sum(Claveles) as Claveles  
    ,sum(Macetas) as Macetas  
    ,sum(Tierra) as Tierra  
    ,sum(Girasoles) as Girasoles  
    ,sum(Hortensia) as Hortensia  
    ,sum(Globos) as Globos  
    ,sum(Tarjetas) as Tarjetas  
    ,sum(Orquideas) as Orquideas  
    ,sum(Carmesi) as Carmesi  
    ,sum(Lirios) as Lirios  
    ,sum(Aurora) as Aurora  
    ,sum(Tulipanes) as Tulipanes  
    ,sum(Liston) as Liston
```

```
FROM [EJERCICIO2].[dbo].[Floristeria_Pais]
```



SQLQuery6.sql - DE...9RD\Roberto (141)) * X SQLQuery5.sql - DE...9RD\Roberto (136) SQLQuery4.sql - DE...9RD\Roberto (133))

```
-- Script para conocer las ventas de cada producto a nivel pais  
SELECT  
    sum(Rosas) as Rosas  
    ,sum(Claveles) as Claveles  
    ,sum(Macetas) as Macetas  
    ,sum(Tierra) as Tierra  
    ,sum(Girasoles) as Girasoles  
    ,sum(Hortensia) as Hortensia  
    ,sum(Globos) as Globos  
    ,sum(Tarjetas) as Tarjetas  
    ,sum(Orquideas) as Orquideas  
    ,sum(Carmesi) as Carmesi  
    ,sum(Lirios) as Lirios  
    ,sum(Aurora) as Aurora  
    ,sum(Tulipanes) as Tulipanes  
    ,sum(Liston) as Liston  
FROM [EJERCICIO2].[dbo].[Floristeria_Pais]
```

	Rosas	Claveles	Macetas	Tierra	Girasoles	Hortensia	Globos	Tarjetas	Orquideas	Carmesi	Lirios	Aurora	Tulipanes	Liston
1	945	733	778	745	787	774	892	779	797	747	795	804	753	975

Concluimos que a nivel de país los productos mas vendidos son “El listo” y “las rosas”, siendo lo menos buscado “Los claveles” y el “carmesí”