INSTITUTO TECNOLÓGICO DE ESTUDIOS SUPERIORES DE OCCIDENTE

Departamento de Electrónica, Sistemas e Informática.

ITESO

Universidad Jesuita de Guadalajara

Proyecto (Fase 1):

Creación de un almacén de datos con un modelo de constelación

Materia:	Almacenes de Datos
Integrantes del equipo:	Alan Daniel Ríos López | 734869 Paulina Flores Sánchez | 745570 Sergio David Elizondo Silva |
Profesor:	Sergio Rubén Aceves Azuara
Fecha:	17 de octubre de 2025

I. Introducción

El presente proyecto tiene como objetivo la creación de un Data Warehouse con datos dimensionales extraídos de una base de datos transaccional (*AutoPartes*). Con base en el DW se busca la implementación de herramientas ETL y de consultas de datos, con el propósito de facilitar el análisis, búsqueda y toma de decisiones empresariales respecto a los datos obtenidos.

Primeramente, un **Data Warehouse** es un sistema que agrega datos de varias fuentes en un único almacén de datos central. Estos sistemas pueden ingerir grandes cantidades de información proveniente de una amplia gama de sistemas de origen, como bases de datos operativas, sistemas transaccionales y plataformas de gestión de relaciones con los clientes (Jonker, A., Holdsworth, J. & Kosinski, M.).

Con ayuda de las herramientas de análisis de servicios con las que cuentan los DW, los usuarios empresariales pueden explorar y analizar estos datos para obtener perspectivas valiosas con respecto a la información disponible. Durante el desarrollo del proyecto se espera construir un modelo de constelación que alimente el DW, permitiendo a los usuarios acceder a estos análisis.

En segundo lugar, los modelos o **diagramas de constelación** son un tipo de esquema de base de datos que consta de múltiples tablas de hechos y de dimensiones compartidas. Las tablas de hechos contienen diferentes niveles de granularidad o perspectivas de los hechos, mientras que las de dimensiones contienen los atributos para unir las tablas de hechos. Se les llama modelo de constelación porque el esquema se asemeja a una constelación, con múltiples estrellas conectadas por dimensiones compartidas (LinkedIn, 2023).

Por otro lado, para construir el poblado de datos del DW se utilizará la herramienta para procesos ETL, **KNIME**.

> KNIME es una herramienta de análisis y ciencia de datos que te permite crear flujos de trabajo de datos de cualquier complejidad con una programación visual de arrastrar y soltar, muy accesible y sin código (Davles, 2024).

En otras palabras, KNIME permite transformar procesos analíticos en otros más complejos, mediante la transformación de datos a un formato más visual y didáctico. No obstante, si se habla de herramientas que faciliten el análisis de la información, se deben mencionar los cubos OLAP y los tableros.

Un **cubo OLAP** (o cubo de procesamiento analítico en línea) puede mostrar y resumir grandes cantidades de datos, al mismo tiempo que permite a los usuarios explorar cualquiera de estos. De este modo, los datos se pueden consolidar, segmentar y analizar según sea necesario para responder una amplia variedad de preguntas relevantes para el área de interés del usuario (Microsoft, 2024).

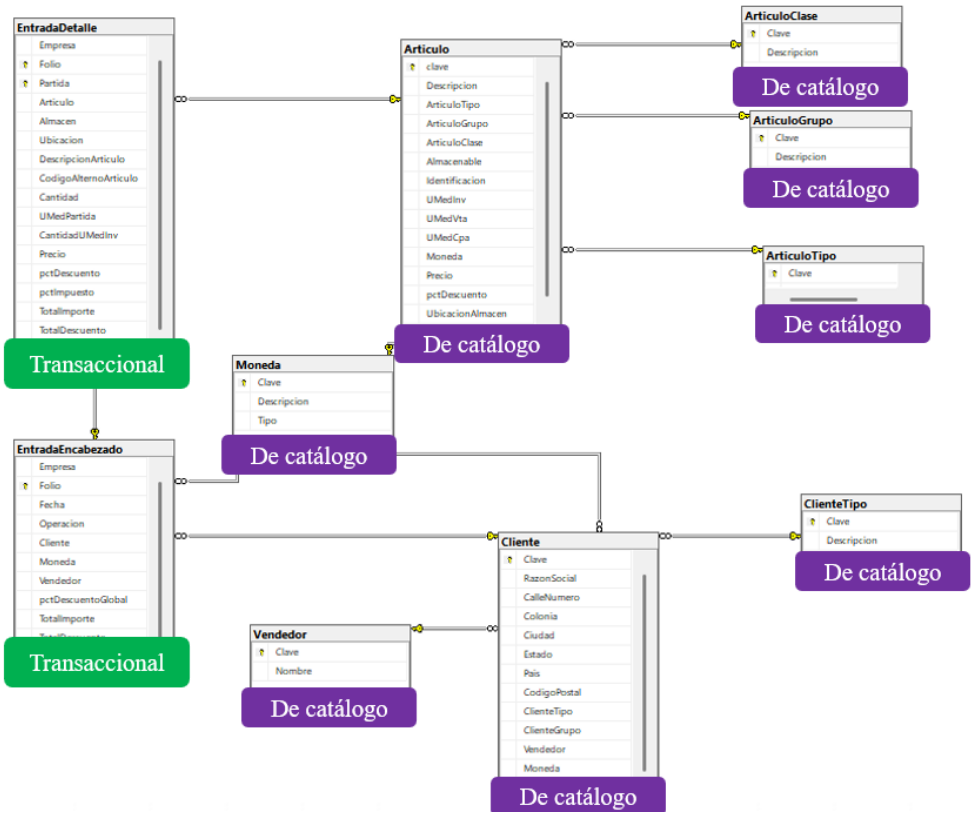
Por último, los **tableros de análisis de datos** son herramientas visuales que permiten concentrar, organizar y presentar la información clave del negocio en un solo lugar. Su objetivo es "ofrecer una vista clara y accionable del desempeño en diferentes áreas. Esto permite tomar decisiones informadas en tiempo real" (Intelisis, 2025).

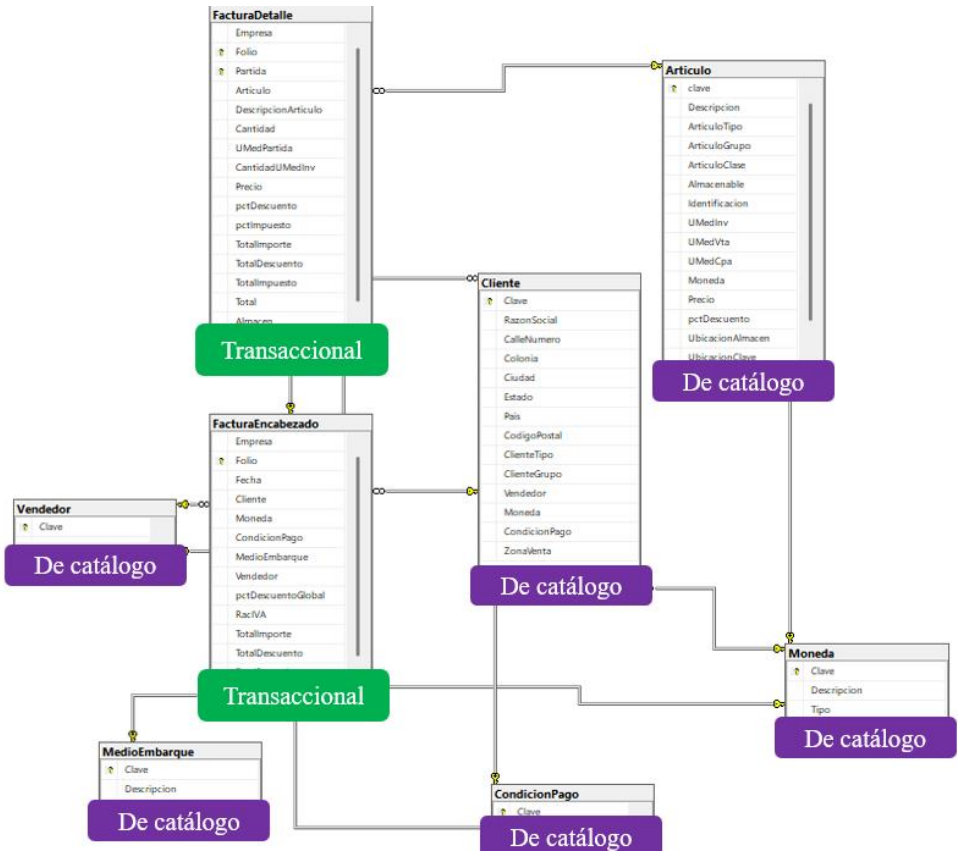
Utilizando las herramientas y modelos mencionados, se cumplirán los objetivos de facilitar el análisis, búsqueda y toma de decisiones empresariales respecto a los datos obtenidos en la base de datos AutoPartes.

II. Base de datos con diagrama de constelación

1. Diagramas de AutoPartes

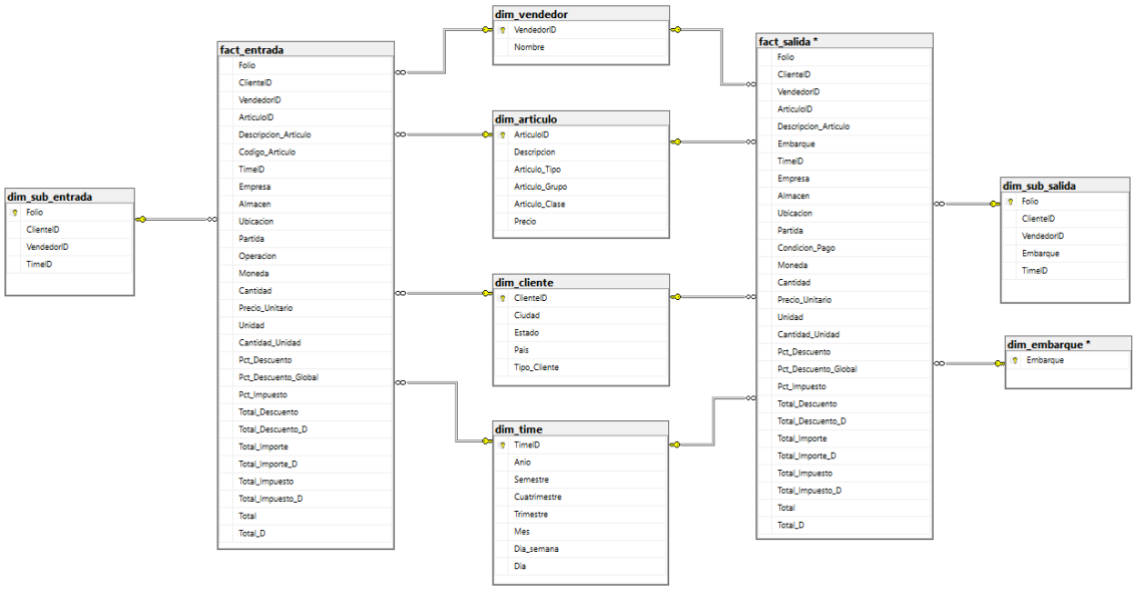
Para comenzar a desarrollar nuestro modelo de constelación, primero se debe categorizar y analizar las tablas de *AutoPartes*. Durante el desarrollo de este proyecto se hará uso de dos: *Entradas* y *Salidas*.


EntradaDetalle
Empresa
Folio
Partida
Articulo
Almacen
Ubicacion
DescripcionArticulo
CodigoAlternoArticulo
Cantidad
UMedPartida
CantidadUMedInv
Precio
pctDescuento
pctImpuesto
TotalImporte
TotalDescuento
Transaccional
Articulo
clave
Descripcion
ArticuloTipo
ArticuloGrupo
ArticuloClase
Almacenable
Identificacion
UMedInv
UMedVta
UMedCpa
Moneda
Precio
pctDescuento
UbicacionAlmacen
De catálogo
ArticuloClase
Clave
Descripcion
De catálogo
ArticuloGrupo
Clave
Descripcion
De catálogo
ArticuloTipo
Clave
De catálogo
Moneda
Clave
Descripcion
Tipo
De catálogo
EntradaEncabezado
Empresa
Folio
Fecha
Operacion
Cliente
Moneda
Vendedor
pctDescuentoGlobal
TotalImporte
Transaccional
Cliente
Clave
RazonSocial
CalleNumero
Colonia
Ciudad
Estado
Pais
CodigoPostal
ClienteTipo
ClienteGrupo
Vendedor
Moneda
De catálogo
ClienteTipo
Clave
Descripcion
De catálogo
Vendedor
Clave
Nombre
De catálogo


FacturaDetalle
Empresa
Folio
Partida
Articulo
DescripcionArticulo
Cantidad
UMedPartida
CantidadUMedInv
Precio
pctDescuento
pctImpuesto
TotalImporte
TotalDescuento
TotalImpuesto
Total
Transaccional
Articulo
clave
Descripcion
ArticuloTipo
ArticuloGrupo
ArticuloClase
Almacenable
Identificacion
UMedInv
UMedVta
UMedCpa
Moneda
Precio
pctDescuento
UbicacionAlmacen
De catálogo
Cliente
Clave
RazonSocial
CalleNumero
Colonia
Ciudad
Estado
Pais
CodigoPostal
ClienteTipo
ClienteGrupo
Vendedor
Moneda
CondicionPago
ZonaVenta
De catálogo
FacturaEncabezado
Empresa
Folio
Fecha
Cliente
Moneda
CondicionPago
MedioEmbarque
Vendedor
pctDescuentoGlobal
RacIVA
TotalImporte
TotalDescuento
Transaccional
Vendedor
Clave
De catálogo
Moneda
Clave
Descripcion
Tipo
De catálogo
MedioEmbarque
Clave
Descripcion
De catálogo
CondicionPago
Clave
De catálogo

2. Diagrama de constelación

Después de la categorización de las tablas de *AutoPartes*, se realiza el diagrama de constelación, el cual contendrá dos tablas de hechos: *fact_entrada* y *fact_salida*, unidas por las tablas dimensionales *dim_articulo*, *dim_cliente*, *dim_time* y *dim_vendedor*. Además, cada tabla de hechos cuenta con su propia tabla dimensional, en donde se guardan los IDs necesarios a la hora de realizar una búsqueda.



3. Poblado de datos en SQL

Una vez construido el diagrama y las tablas, el siguiente paso es trabajar en el poblado de datos. Dentro de este, necesitamos los stored procedures de cada tabla, los cuales se encuentran en los anexos del documento. Por lo que, el código para la ejecución del poblado de datos es el siguiente.

```
-- Poblado de las tablas

ALTER PROCEDURE dbo.poblar_es
AS BEGIN
    -- Eliminacion de las tablas fact
    DELETE [dbo].[fact_entrada];
    DELETE [dbo].[fact_salida];

    -- Eliminacion de las tablas dim
    DELETE [dbo].[dim_articulo];
```

```
    DELETE [dbo].[dim_cliente];
    DELETE [dbo].[dim_entrada];
    DELETE [dbo].[dim_salida];
    DELETE [dbo].[dim_time];
    DELETE [dbo].[dim_vendedor];

    -- Datos de la tabla dim_articulo
    INSERT INTO [dbo].[dim_articulo]
    EXEC [dbo].[sp_dim_articulo]
    PRINT 'Tabla dim_articulo poblada!'

    -- Datos de la tabla dim_cliente
    INSERT INTO [dbo].[dim_cliente]
    EXEC [dbo].[sp_dim_cliente]
    PRINT 'Tabla dim_cliente poblada!'

    -- Datos de la tabla dim_entrada
    INSERT INTO [dbo].[dim_entrada]
    EXEC [dbo].[sp_dim_entrada]
    PRINT 'Tabla dim_entrada poblada!'

    -- Datos de la tabla dim_salida
    INSERT INTO [dbo].[dim_salida]
    EXEC [dbo].[sp_dim_salida]
    PRINT 'Tabla dim_salida poblada!'

    -- Datos de la tabla dim_time
    INSERT INTO [dbo].[dim_time]
    EXEC [dbo].[sp_dim_time]
    PRINT 'Tabla dim_time poblada!'

    -- Datos de la tabla dim_vendedor
    INSERT INTO [dbo].[dim_vendedor]
    EXEC [dbo].[sp_dim_vendedor]
    PRINT 'Tabla dim_time poblada!'

    -- Insercion de un registro "no valido" dentro de dim_vendedor
    INSERT INTO [dbo].[dim_vendedor] (VendedorID, Nombre)
    SELECT 'NO PROPORCIONADO', 'NO PROPORCIONADO'
    WHERE NOT EXISTS (SELECT 1 FROM [dbo].[dim_vendedor] WHERE VendedorID
= 'NO PROPORCIONADO');

    -- Insercion de un registro "no valido" dentro de dim_cliente
    INSERT INTO [dbo].[dim_cliente] (ClienteID, Ciudad, Estado, Pais,
Tipo_Cliente)
```

```
    SELECT 'NO PROPORCIONADO', 'NO PROPORCIONADO', 'NO PROPORCIONADO',
'NO PROPORCIONADO', 'NO PROPORCIONADO'
    WHERE NOT EXISTS (SELECT 1 FROM [dbo].[dim_cliente] WHERE ClienteID =
'NO PROPORCIONADO');

    -- Datos de la tabla fact_entrada
    INSERT INTO [dbo].[fact_entrada]
    EXEC [dbo].[sp_fact_entrada]
    PRINT 'Tabla fact_entrada poblada!'

    -- Datos de la tabla fact_salida
    INSERT INTO [dbo].[fact_salida]
    EXEC [dbo].[sp_fact_salida]
    PRINT 'Tabla fact_salida poblada!'

    PRINT 'Todas las tablas pobladas!!'
END;
```

Con el poblado de datos ya creado, se verifica la información obtenida mediante casos de prueba, en los que se comparan los resultados entre la base de datos *AutoPartes* y *Proyecto_ES*.

A. Total de artículos de Chevrolet dentro de entradas con los folios CF000077 y A003938

Con el fin de conseguir la información requerida, se ejecuta este Query dentro de *AutoPartes*.

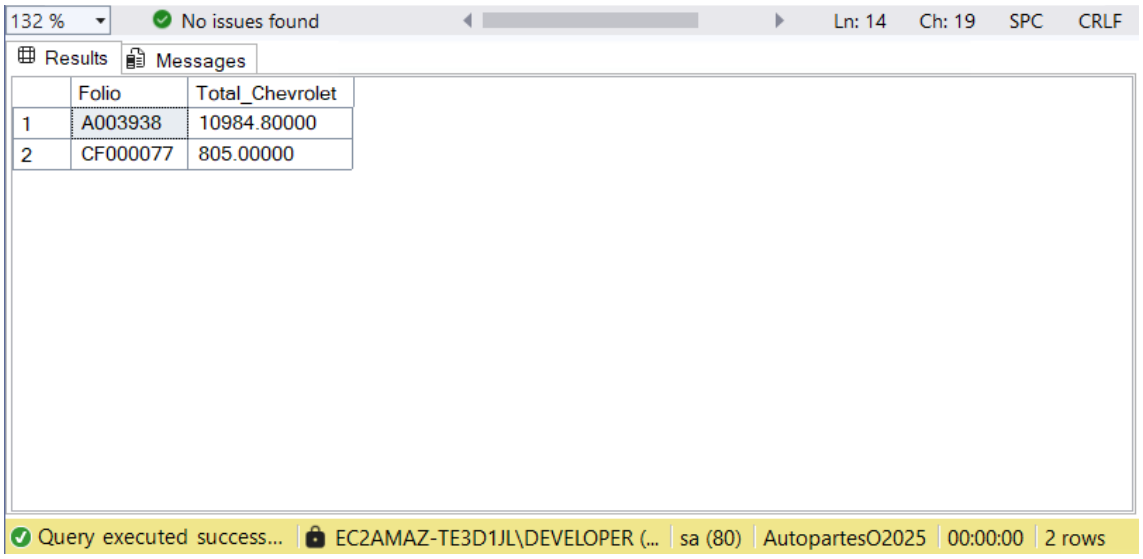
```
SELECT
    e.Folio,
    SUM(d.Total) AS Total_Chevrolet
FROM [AutopartesO2025].[dbo].[EntradaEncabezado] e

LEFT JOIN [AutopartesO2025].[dbo].[EntradaDetalle] d
    ON e.Folio = d.Folio
LEFT JOIN [AutopartesO2025].[dbo].[Articulo] a
    ON d.Articulo = a.Clave
LEFT JOIN [AutopartesO2025].[dbo].[ArticuloGrupo] ag
    ON a.ArticuloGrupo = ag.Clave

WHERE
    UPPER(ISNULL(ag.Descripcion, '')) LIKE '%CHEVROLET%'
```

```
    AND e.Folio IN ('CF000077', 'A003938')
GROUP BY e.Folio;
```

El cual da como resultado la siguiente tabla.



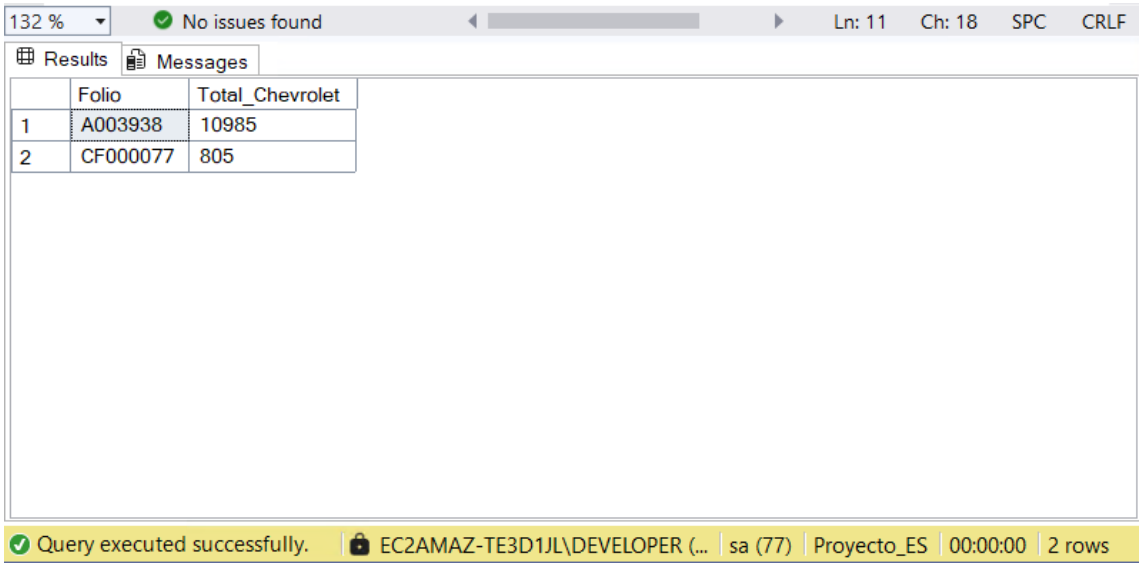
Por otro lado, dentro de *Proyecto_ES* se ejecuta un Query parecido.

```
SELECT
    e.Folio,
    SUM(e.Total_D) AS Total_Chevrolet
FROM [Proyecto_ES].[dbo].[fact_entrada] e

LEFT JOIN [Proyecto_ES].[dbo].[dim_articulo] a
    ON e.ArticuloID = a.ArticuloID

WHERE
    UPPER(LTRIM(RTRIM(a.Articulo_Grupo))) = 'CHEVROLET'
    AND e.Folio IN ('CF000077', 'A003938')
GROUP BY e.Folio;
```

El cual retorna los mismos resultados que en *AutoPartes*.



B. Cantidad de DEFENSA DEL GOLF GTI dentro de la entrada del folio A003933

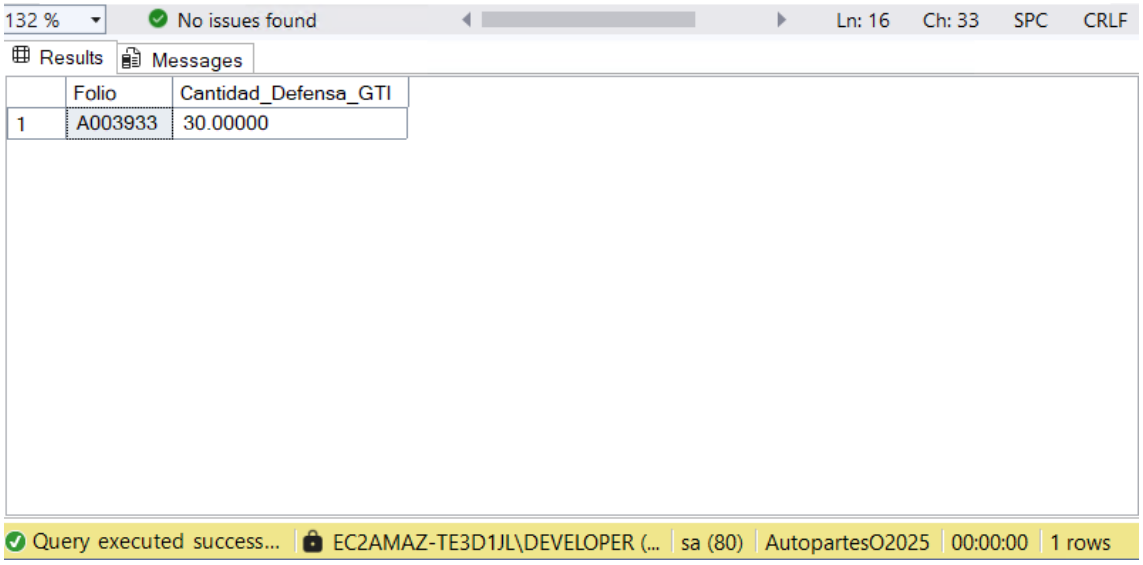
Para obtener la información requerida de *AutoPartes*, se ejecuta el siguiente Query.

```
SELECT
    UPPER(TRIM(e.[Folio])) AS Folio,
    SUM(d.[Cantidad]) AS Cantidad_Defensa_GTI
FROM [AutoPartesO2025].[dbo].[EntradaEncabezado] e

LEFT JOIN [AutoPartesO2025].[dbo].[EntradaDetalle] d
    ON e.[Folio] = d.[Folio]
LEFT JOIN [AutoPartesO2025].[dbo].[Articulo] a
    ON d.[Articulo] = a.[Clave]

WHERE
    UPPER(LTRIM(RTRIM(a.[Descripcion]))) LIKE '%DEFENSA%'
    AND UPPER(LTRIM(RTRIM(a.[Descripcion]))) LIKE '%GOLF%'
    AND UPPER(LTRIM(RTRIM(a.[Descripcion]))) LIKE '%GTI%'
    AND UPPER(TRIM(e.[Folio])) = 'A003933'
GROUP BY UPPER(TRIM(e.[Folio]));
```

El cual da como resultado 30.



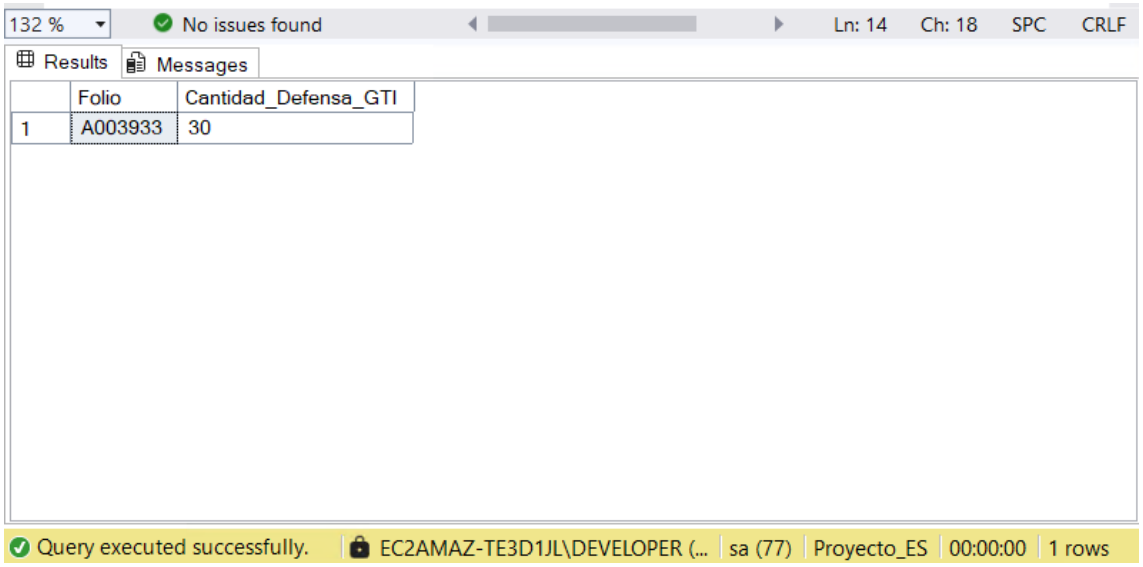
Por otro lado, dentro de *Proyecto_ES* se ejecuta un Query parecido.

```sql
SELECT
    e.Folio,
    SUM(e.Cantidad) AS Cantidad_Defensa_GTI
FROM [Proyecto_ES].[dbo].[fact_entrada] e

LEFT JOIN [Proyecto_ES].[dbo].[dim_articulo] a
    ON e.ArticuloID = a.ArticuloID

WHERE
    UPPER(LTRIM(RTRIM(a.Descripcion))) LIKE '%DEFENSA%'
    AND UPPER(LTRIM(RTRIM(a.Descripcion))) LIKE '%GOLF%'
    AND UPPER(LTRIM(RTRIM(a.Descripcion))) LIKE '%GTI%'
    AND e.Folio = 'A003933'
GROUP BY e.Folio;
```

El cual devuelve la misma cantidad de dicho artículo: 30.



C. Cantidad de PARRILLAS de salida con el folio A0020915

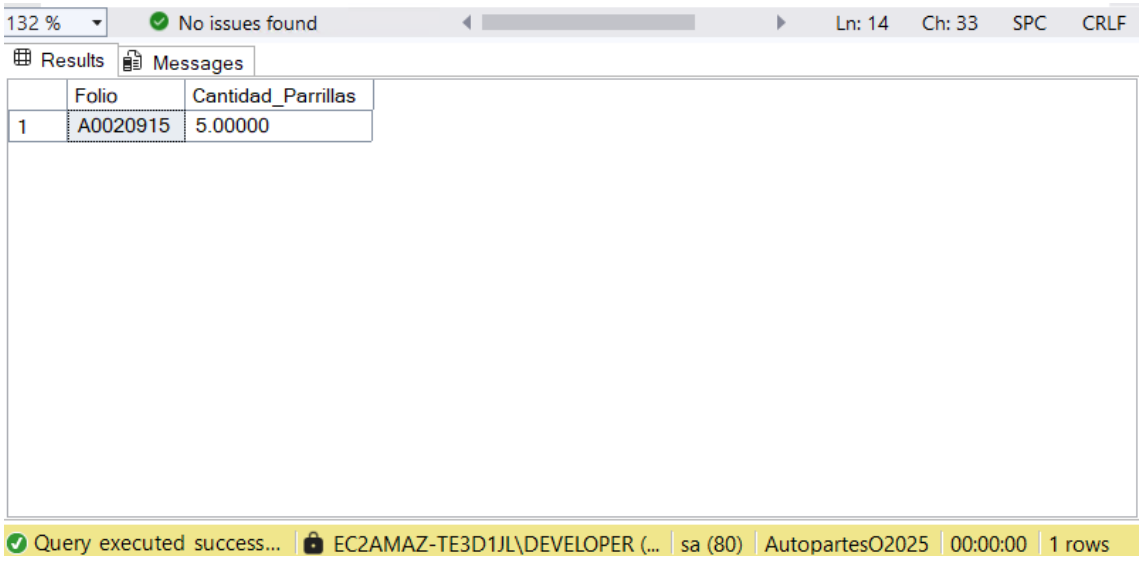
Con el fin de conseguir la información requerida, se ejecuta este Query dentro de *AutoPartes*.

```
SELECT
    UPPER(TRIM(e.[Folio])) AS Folio,
    SUM(d.[Cantidad]) AS Cantidad_Parrillas
FROM [AutoPartesO2025].[dbo].[SalidaEncabezado] e

LEFT JOIN [AutoPartesO2025].[dbo].[SalidaDetalle] d
    ON e.[Folio] = d.[Folio]
LEFT JOIN [AutoPartesO2025].[dbo].[Articulo] a
    ON d.[Articulo] = a.[Clave]

WHERE
    UPPER(LTRIM(RTRIM(a.[Descripcion]))) LIKE '%PARRILLA%'
    AND UPPER(TRIM(e.[Folio])) = 'A0020915'
GROUP BY UPPER(TRIM(e.[Folio]));
```

El cual da como resultado 5 parrillas.



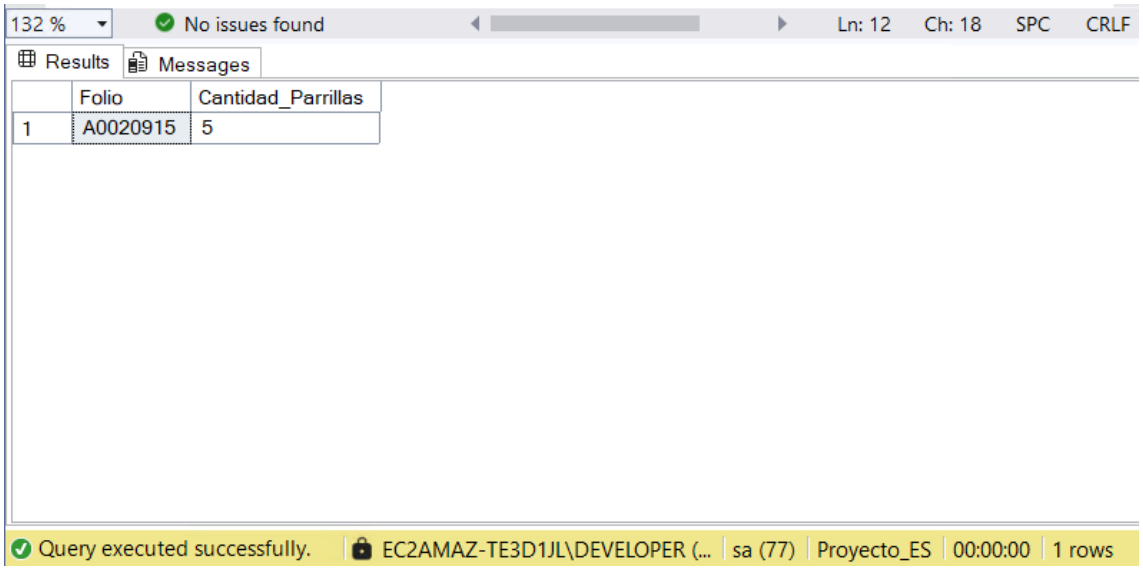
Por otro lado, dentro de *Proyecto_ES* se ejecuta un Query parecido.

```
SELECT
    e.Folio,
    SUM(e.Cantidad) AS Cantidad_Parrillas
FROM [Proyecto_ES].[dbo].[fact_salida] e

LEFT JOIN [Proyecto_ES].[dbo].[dim_articulo] a
    ON e.ArticuloID = a.ArticuloID

WHERE
    UPPER(LTRIM(RTRIM(a.Descripcion))) LIKE '%PARRILLA%'
    AND e.Folio = 'A0020915'
GROUP BY e.Folio;
```

En donde se obtiene como resultado las 5 parrillas dadas en *AutoPartes*.



D. Total en pesos y cantidad de artículos de salida en Ford con los folios A0020927, A0020960 y SA0029433

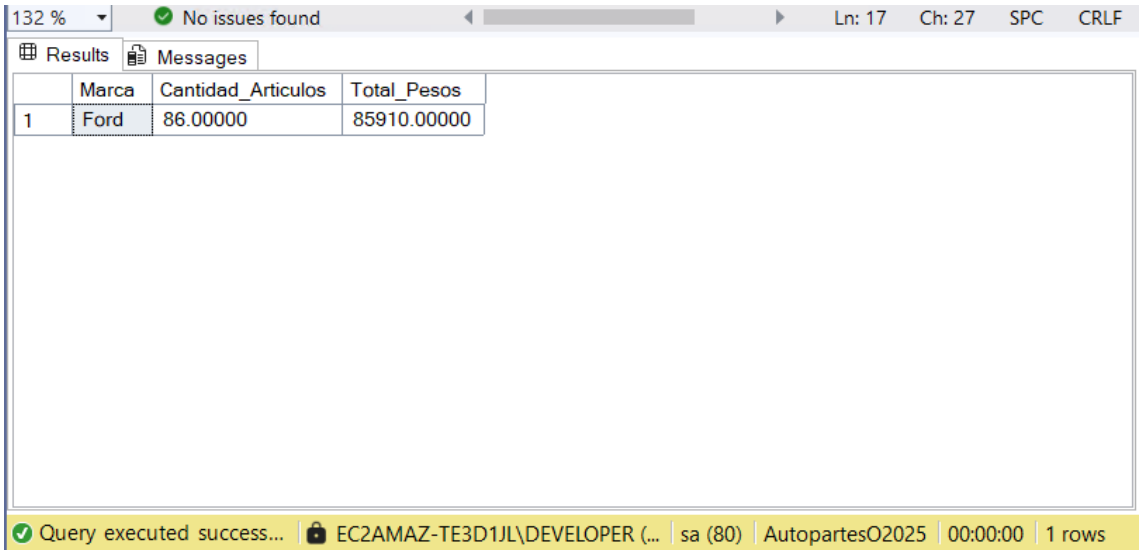
Para obtener la información requerida de *AutoPartes*, se ejecuta el siguiente Query.

```
SELECT
    ag.[Descripcion] AS Marca,
    SUM(d.[Cantidad]) AS Cantidad_Articulos,
    SUM(d.[TotalImporte]) AS Total_Pesos
FROM [AutoPartesO2025].[dbo].[SalidaEncabezado] e

LEFT JOIN [AutoPartesO2025].[dbo].[SalidaDetalle] d
    ON e.[Folio] = d.[Folio]
LEFT JOIN [AutoPartesO2025].[dbo].[Articulo] a
    ON d.[Articulo] = a.[Clave]
LEFT JOIN [AutopartesO2025].[dbo].[ArticuloGrupo] ag
    ON a.[ArticuloGrupo] = ag.[Clave]

WHERE
    UPPER(LTRIM(RTRIM(ag.[Descripcion]))) = 'FORD'
    AND e.[Folio] IN ('A0020927', 'A0020960', 'SA0029433')
GROUP BY ag.[Descripcion];
```

De este Query obtenemos la siguiente tabla como respuesta.



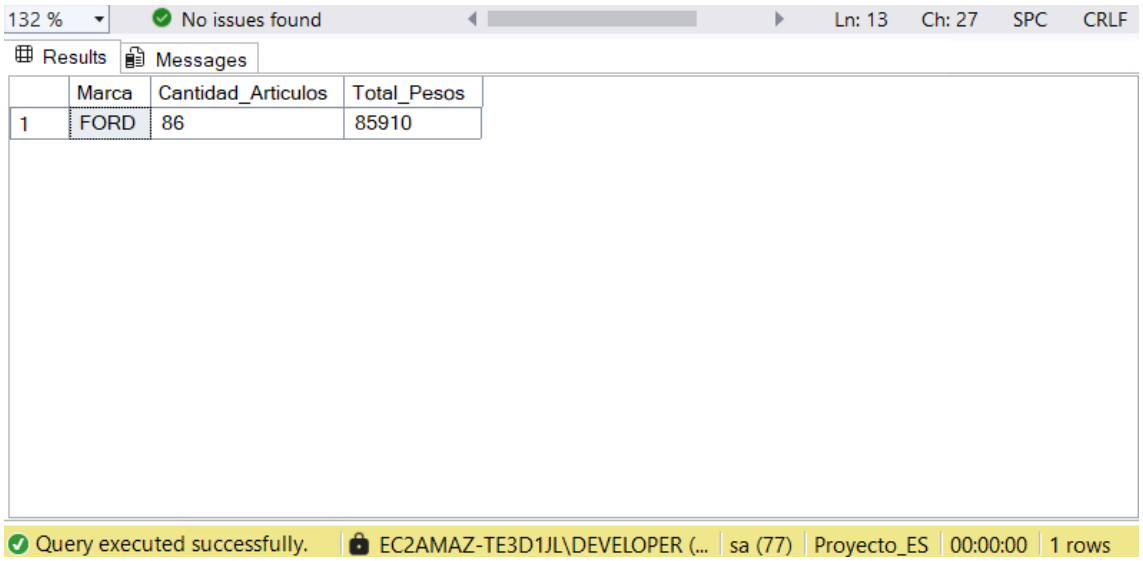
Por otro lado, dentro de *Proyecto_ES* se ejecuta un Query parecido.

```
SELECT
    a.Articulo_Grupo AS Marca,
    SUM(s.Cantidad) AS Cantidad_Articulos,
    SUM(s.Total_Importe_D) AS Total_Pesos
FROM [Proyecto_ES].[dbo].[fact_salida] s

LEFT JOIN [Proyecto_ES].[dbo].[dim_articulo] a
    ON s.ArticuloID = a.ArticuloID

WHERE
    UPPER(LTRIM(RTRIM(a.Articulo_Grupo))) = 'FORD'
    AND s.Folio IN ('A0020927', 'A0020960', 'SA0029433')
GROUP BY a.Articulo_Grupo;
```

En donde se obtiene una tabla con las mismas características que en *AutoPartes*.

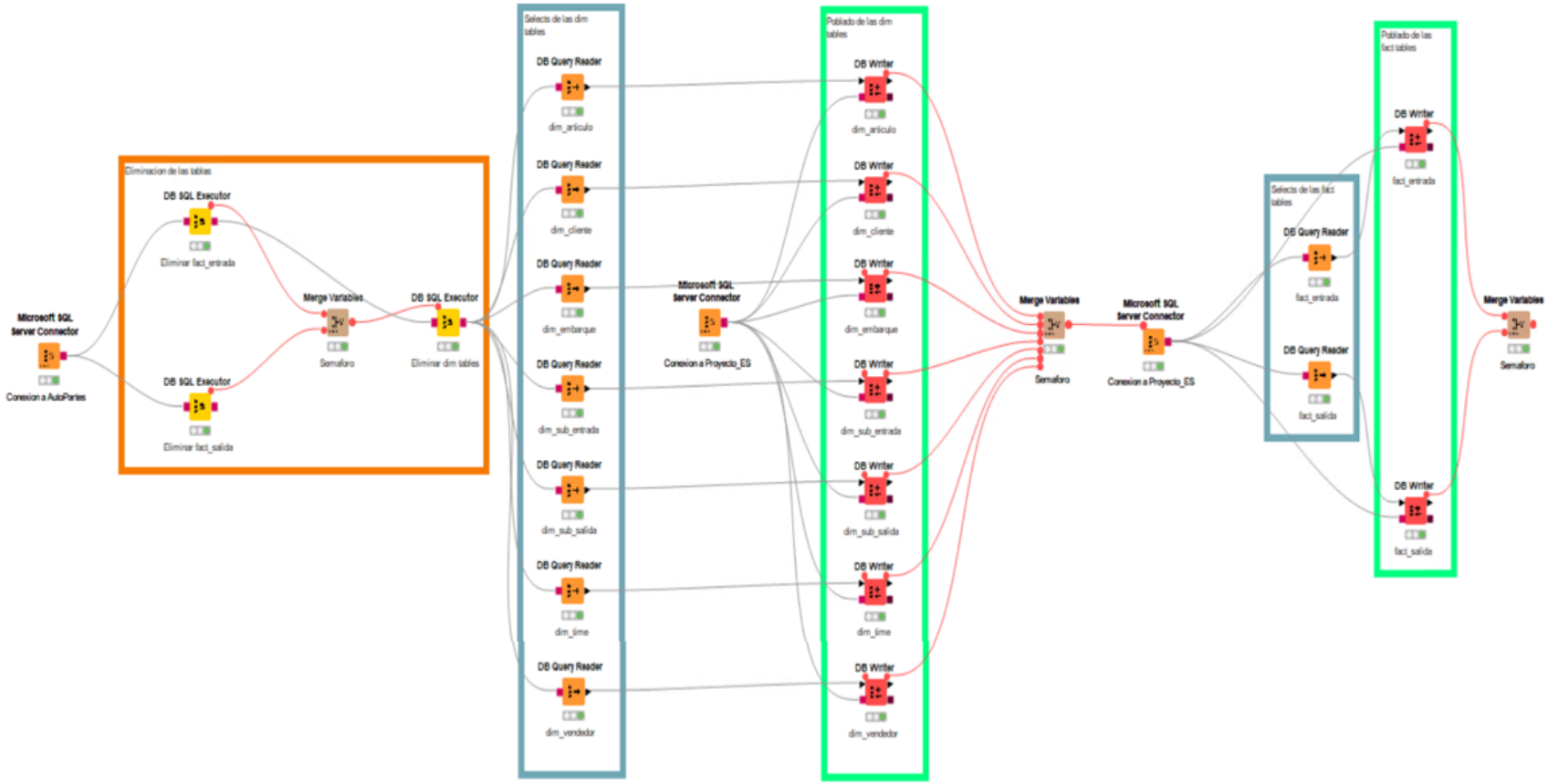


4. Poblado de datos en ETL

Después de contar con el poblado de datos dentro de SQL, el siguiente paso es hacerlo dentro de KNIME. Dentro del nuevo flujo de trabajo se utilizarán los siguientes tipos de nodos para una ejecución eficiente del poblado:

- ***Microsoft SQL Server Connector:*** crea una conexión JDBC a la base de datos seleccionada.
- ***DB SQL Executor:*** permite la ejecución de queries SQL personalizados.
- ***DB Query Reader:*** ejecuta queries de lectura en SQL y muestra los resultados en una tabla.
- ***DB Writer:*** crea y agrega información a las columnas seleccionadas dentro de una tabla dada.
- ***Merge Variables:*** fusiona diferentes nodos en uno solo, continuando con el proceso solo cuando todos los nodos hayan sido ejecutados correctamente.

El flujo final para el poblado de datos en KNIME es el siguiente:

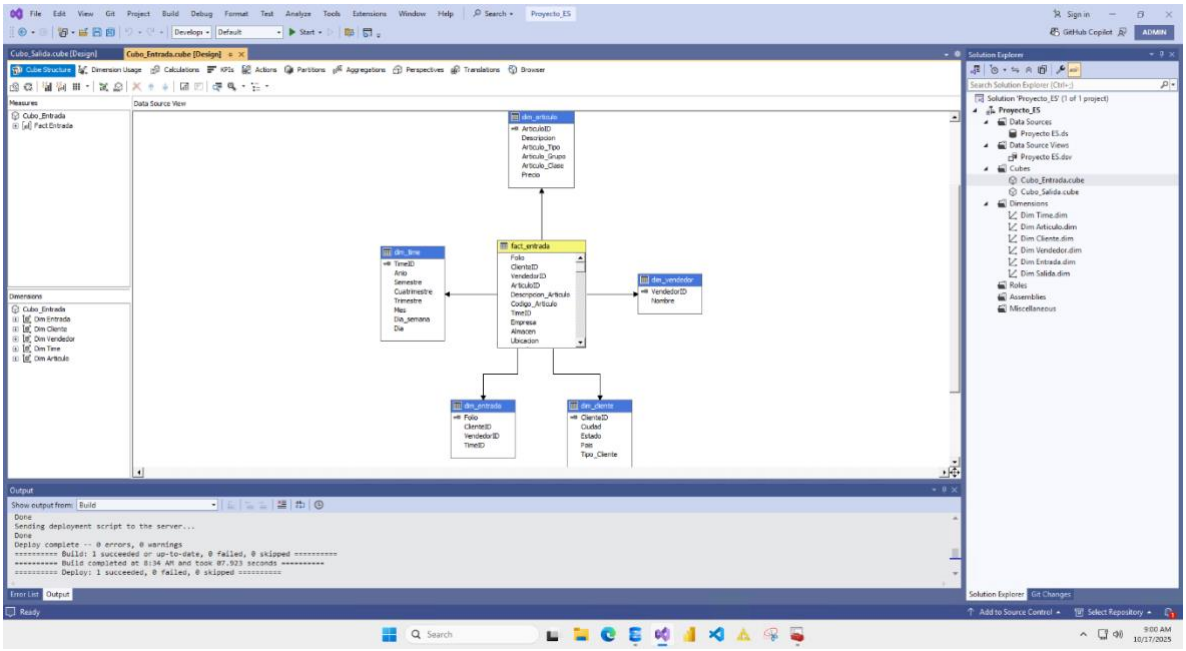

Microsoft SQL Server Connector
Conexion a AutoPartes
Eliminacion de las tablas
DB SQL Executor
Eliminar fact_entrada
DB SQL Executor
Eliminar fact_salida
Merge Variables
Semaforo
DB SQL Executor
Eliminar dim tables
Selects de las dim tables
DB Query Reader
dim_articulo
DB Query Reader
dim_cliente
DB Query Reader
dim_embarque
DB Query Reader
dim_sub_entrada
DB Query Reader
dim_sub_salida
DB Query Reader
dim_time
DB Query Reader
dim_vendedor
Microsoft SQL Server Connector
Conexion a Proyecto_ES
Poblado de las dim tables
DB Writer
dim_articulo
DB Writer
dim_cliente
DB Writer
dim_embarque
DB Writer
dim_sub_entrada
DB Writer
dim_sub_salida
DB Writer
dim_time
DB Writer
dim_vendedor
Merge Variables
Semaforo
Microsoft SQL Server Connector
Conexion a Proyecto_ES
Selects de las fact tables
DB Query Reader
fact_entrada
DB Query Reader
fact_salida
Poblado de las fact tables
DB Writer
fact_entrada
DB Writer
fact_salida
Merge Variables
Semaforo

Este nuevo flujo de trabajo fue construido en base al poblado de SQL, por lo que los pasos para su ejecución son:

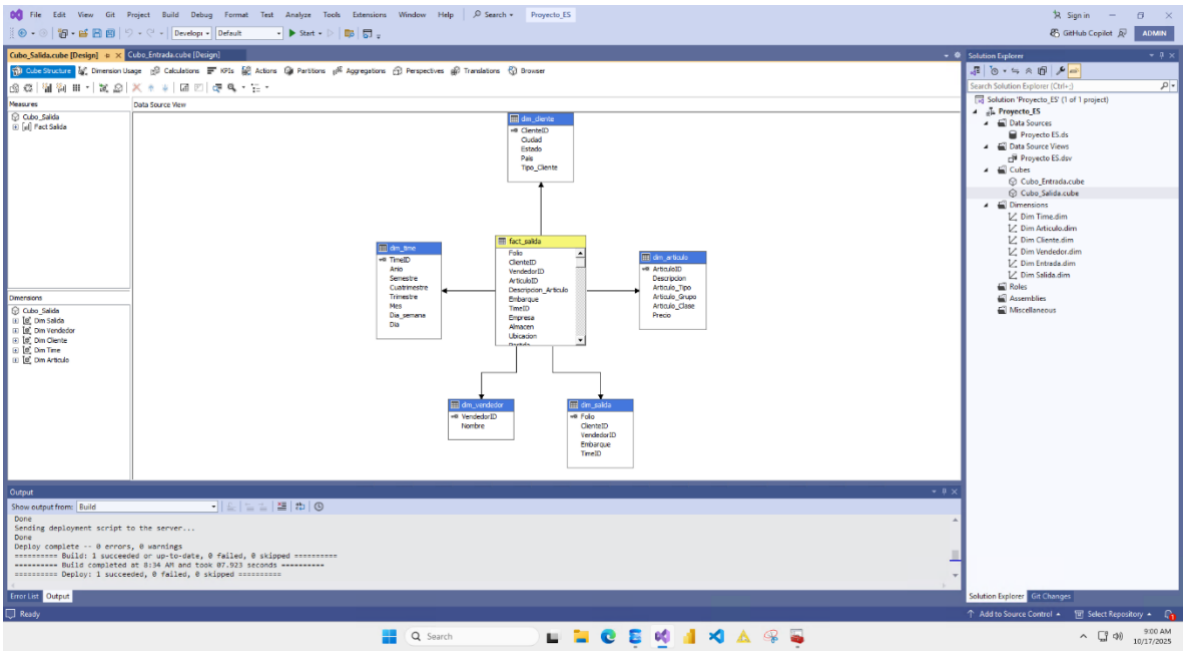
a) Eliminar las tablas de hechos (*fact_entrada* y *fact_salida*).
b) Eliminar las tablas dimensionales (*dim_articulo*, *dim_cliente*, *dim_entrada*, *dim_salida*, *dim_time* y *dim_vendedor*).
c) Poblar las tablas dimensionales mediante selects a la base de datos original (*AutoPartes*).
d) Poblar las tablas de hechos mediante selects a la base de datos original (*AutoPartes*).

5. Creación de cubos

Cubo entrada

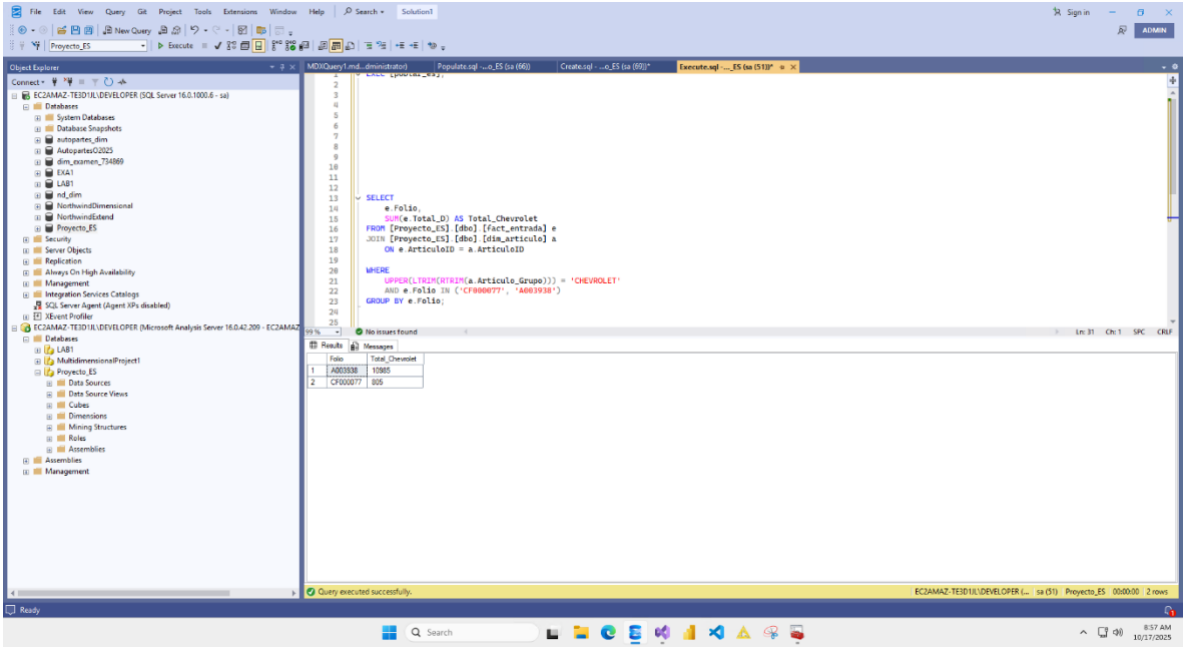


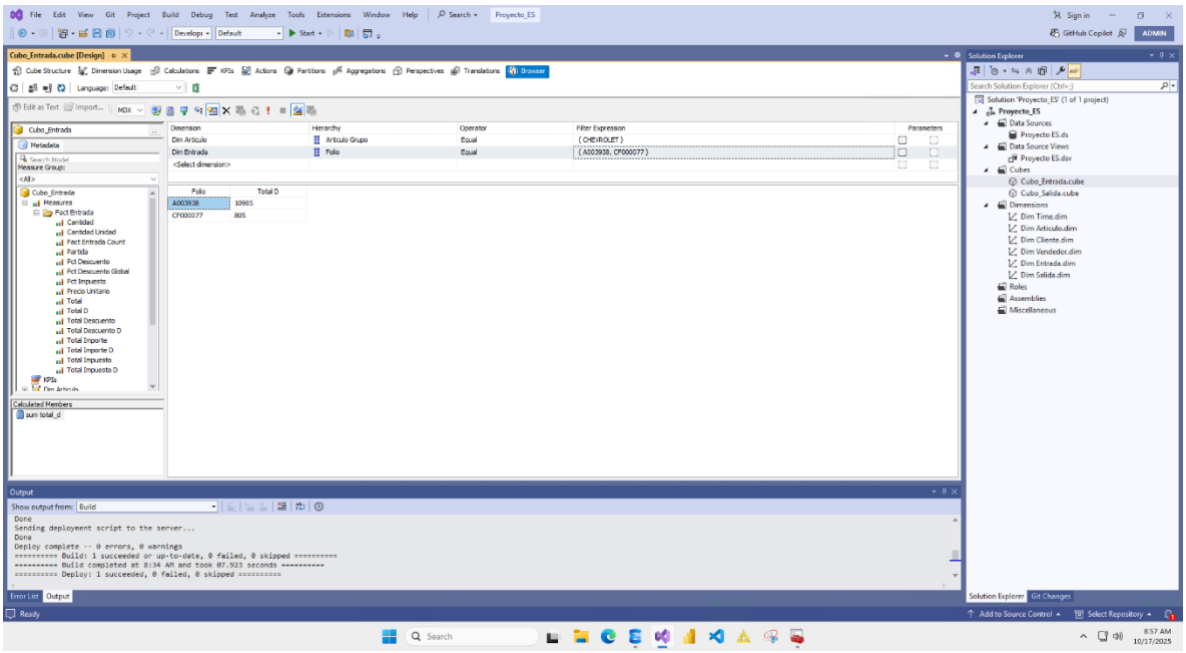
Cubo salida



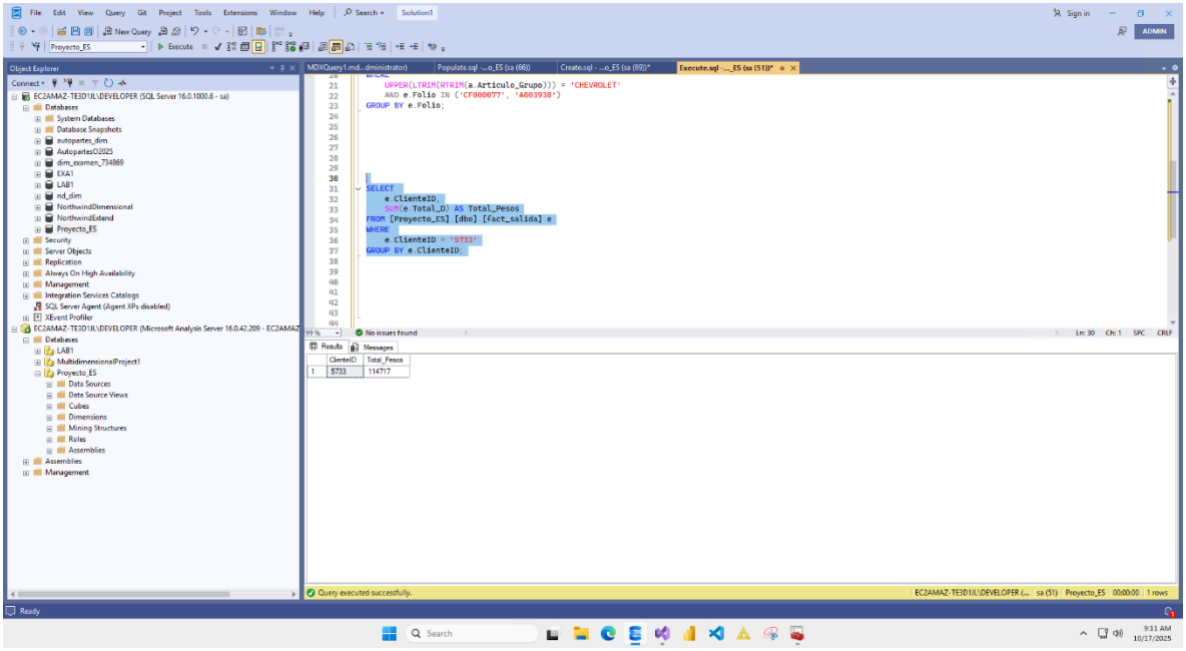
6. Comparación de Cubo con Datawarehouse

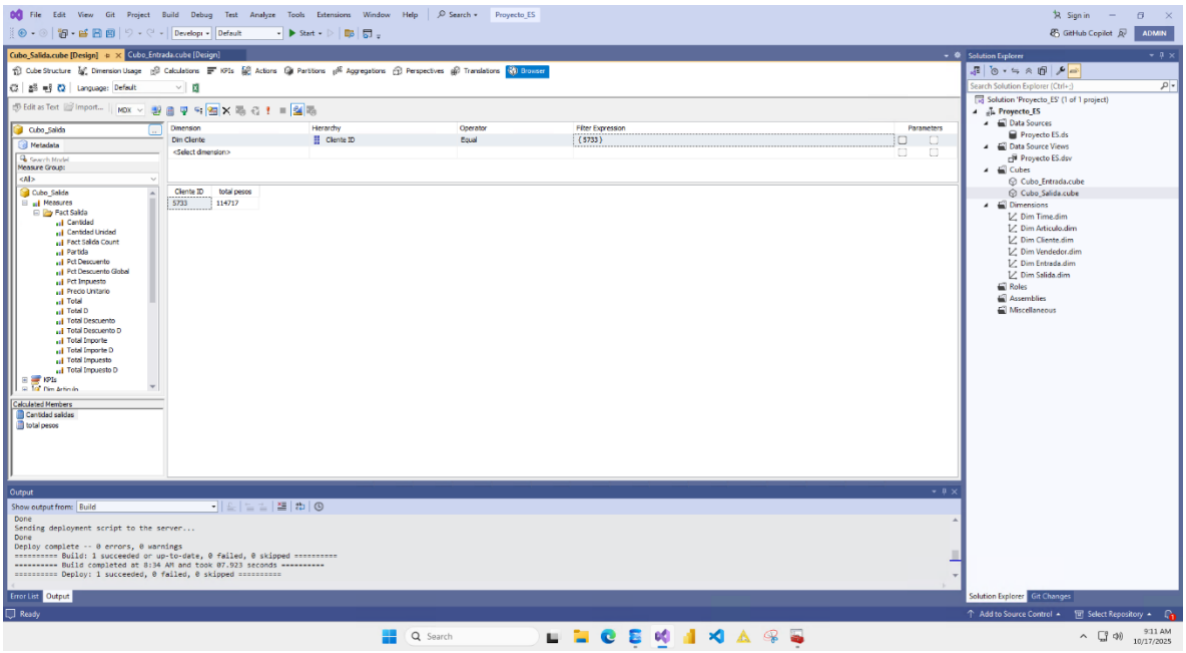
Total de artículos de Chevrloten con folios CF00077 y A003938





Total de pesos del cliente 5733

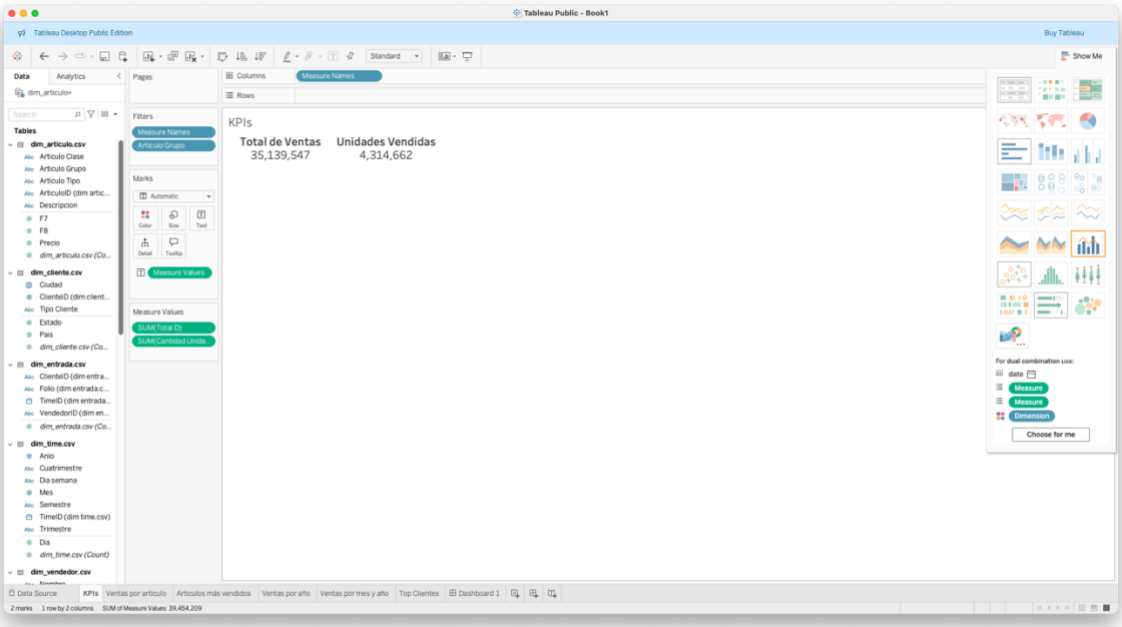




7. Tableros

Dentro de Tableau, se crearon n hojas de trabajo, donde cada una representa una consulta valiosa a nuestra base de datos.

Entradas



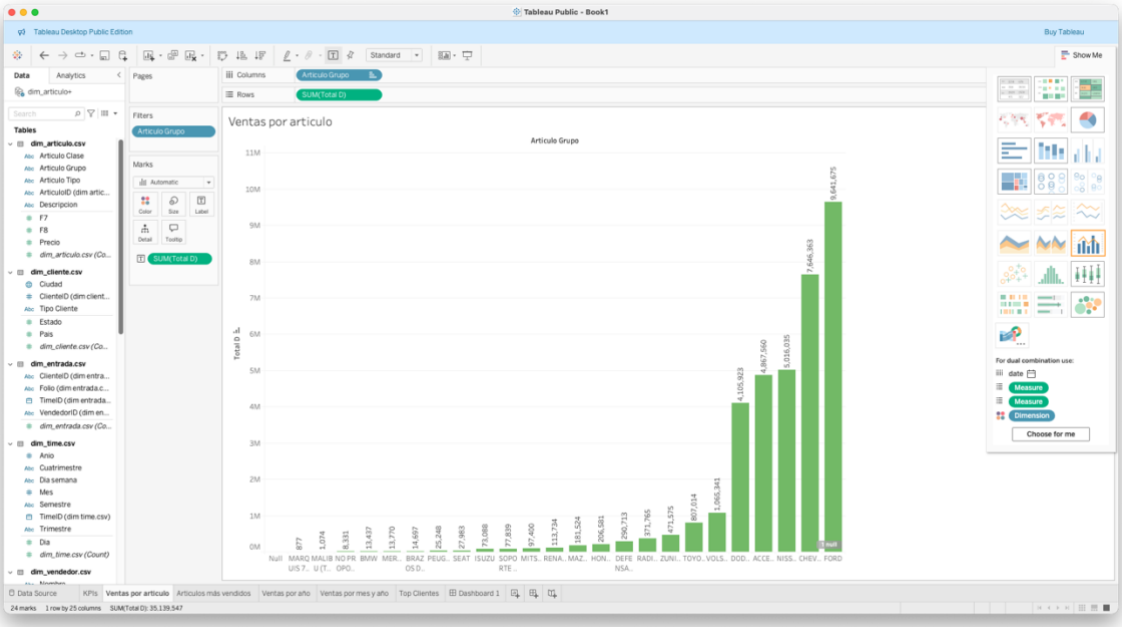


Tableau Public - Book1
Tableau Desktop Public Edition
Buy Tableau
Show Me
Data
Analytics
Pages
Filters
Marks
Columns
Articulo Grupo
Rows
SUM(Total D)
Ventas por articulo
Articulo Grupo
Total D
11M
10M
9M
8M
7M
6M
5M
4M
3M
2M
1M
0M
Null
MARQUIS 7..
MALIBU (T..
NO PROPO..
BMW
MER..
BRAZOS D..
PEUG..
SEAT
ISUZU
SOPORTE ..
MITS..
RENA..
MAZ..
HON..
DEFENSA..
RADI..
ZUNI..
TOYO..
VOLS..
DOD..
ACCE..
NISS..
CHEV..
FORD
Tables
dim_articulo.csv
dim_cliente.csv
dim_entrada.csv
dim_time.csv
dim_vendedor.csv
For dual combination use:
date
Measure
Measure
Dimension
Choose for me
Data Source
KPIs
Ventas por articulo
Articulos más vendidos
Ventas por año
Ventas por mes y año
Top Clientes
Dashboard 1
24 marks
1 row by 25 columns
SUM(Total D): 35,139,547

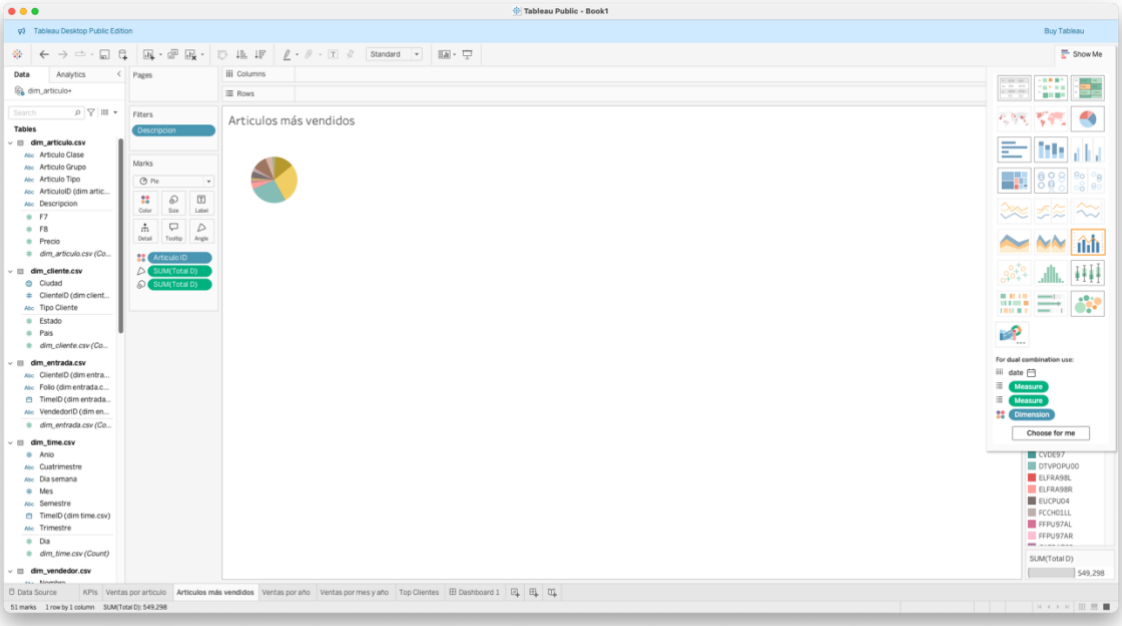


Tableau Public - Book1
Tableau Desktop Public Edition
Buy Tableau
Show Me
Data
Analytics
Pages
Filters
Marks
Columns
Rows
Descripcion
Articulos más vendidos
Articulo ID
SUM(Total D)
SUM(Total D)
Tables
dim_articulo.csv
dim_cliente.csv
dim_entrada.csv
dim_time.csv
dim_vendedor.csv
For dual combination use:
date
Measure
Measure
Dimension
Choose for me
CYDE97
DTVPOPU00
ELFRA98L
ELFRA98R
EUCPU04
FCCH01LL
FFPU97AL
FFPU97AR
SUM(Total D)
549,298
Data Source
KPIs
Ventas por articulo
Articulos más vendidos
Ventas por año
Ventas por mes y año
Top Clientes
Dashboard 1
51 marks
1 row by 1 column
SUM(Total D): 549,298

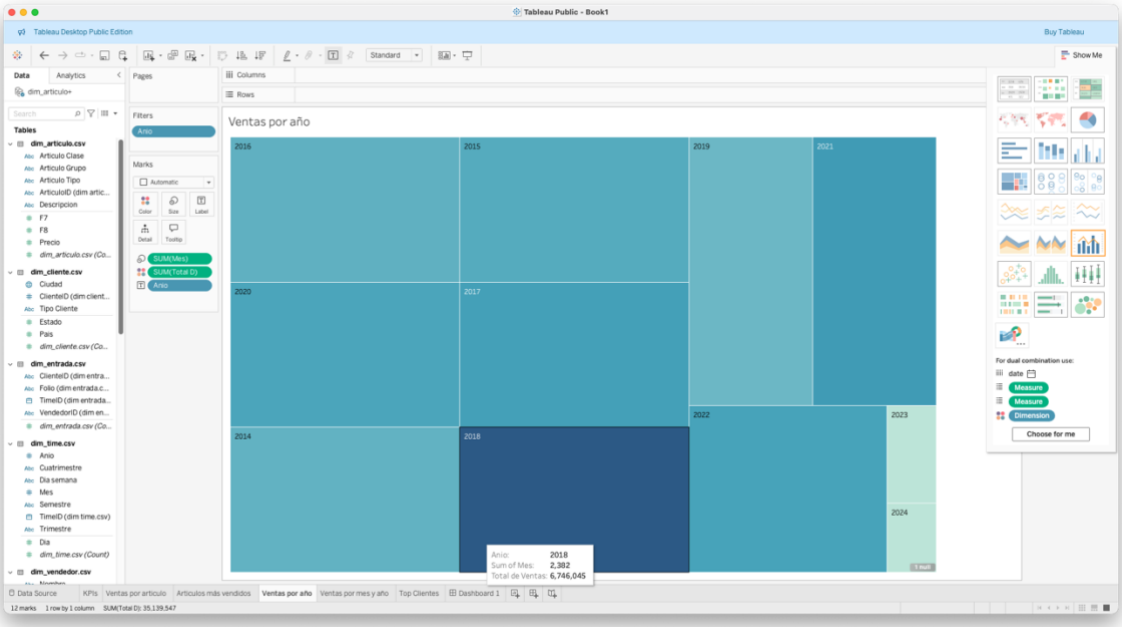


Tableau Public - Book1
Tableau Desktop Public Edition
Buy Tableau
Show Me
Data
Analytics
Pages
Filters
Anio
Marks
Columns
Rows
Ventas por año
2016
2015
2019
2021
2020
2017
2022
2023
2014
2018
2024
Anio: 2018
Sum of Mes: 2,382
Total de Ventas: 6,746,045
For dual combination use:
date
Measure
Measure
Dimension
Choose for me
Data Source
KPIs
Ventas por articulo
Articulos mas vendidos
Ventas por año
Ventas por mes y año
Top Clientes
Dashboard 1
12 marks
1 row by 1 column
SUM(Total D): 35,139,547

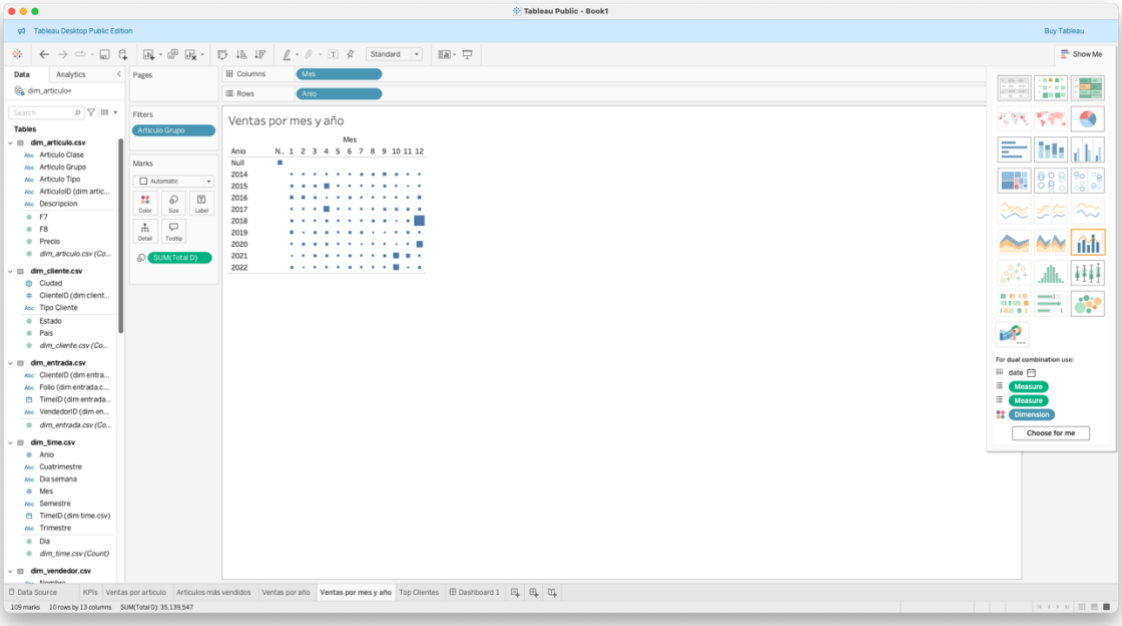
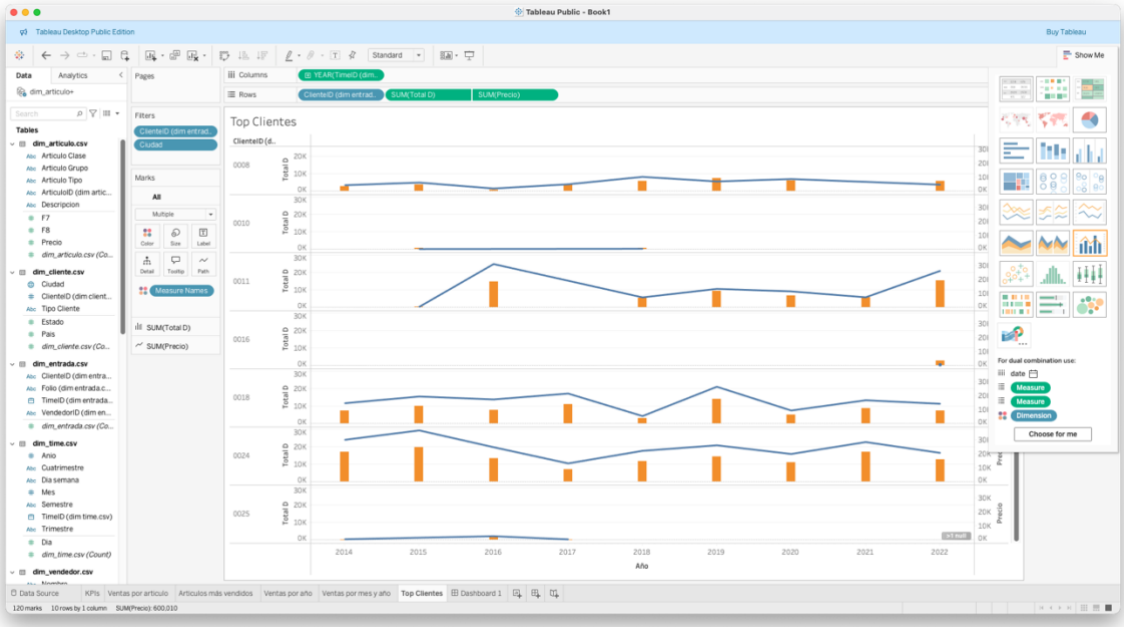


Tableau Public - Book1
Tableau Desktop Public Edition
Buy Tableau
Show Me
Data
Analytics
Pages
Filters
Articulo Grupo
Marks
Columns
Mes
Rows
Anio
Ventas por mes y año
Mes
Anio
Null
2014
2015
2016
2017
2018
2019
2020
2021
2022
For dual combination use:
date
Measure
Measure
Dimension
Choose for me
Data Source
KPIs
Ventas por articulo
Articulos mas vendidos
Ventas por año
Ventas por mes y año
Top Clientes
Dashboard 1
109 marks
10 rows by 13 columns
SUM(Total D): 35,139,547



Tablero Salidas

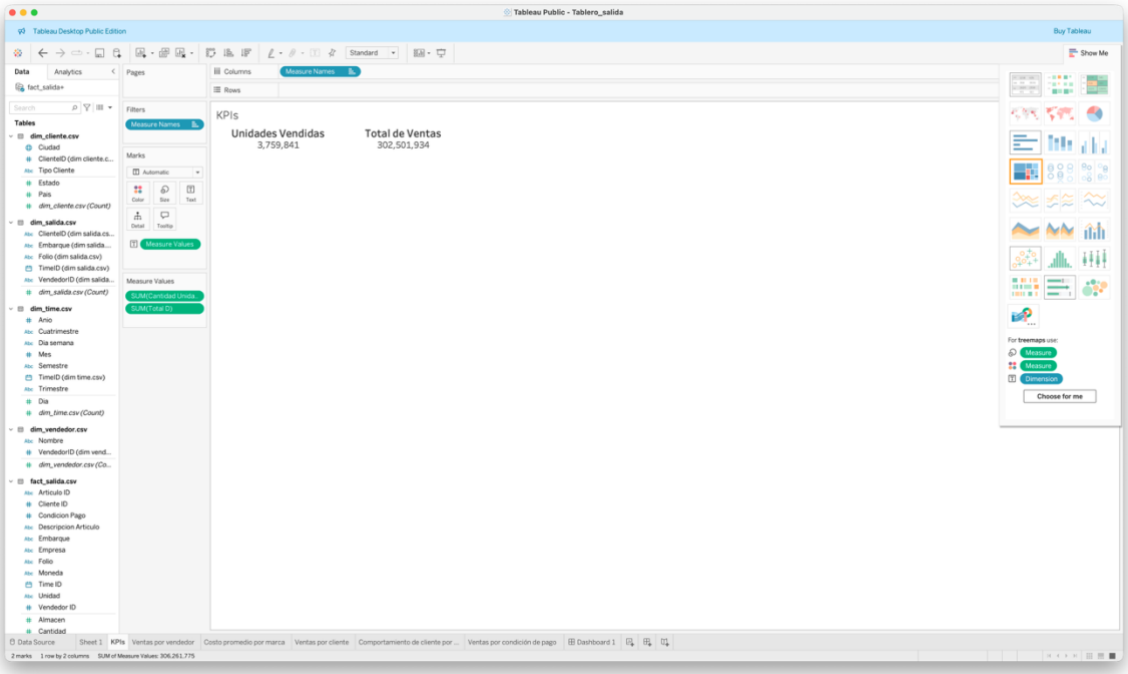


Tableau Public - Tablero_salida
KPIs
Unidades Vendidas
3,759,841
Total de Ventas
302,501,934

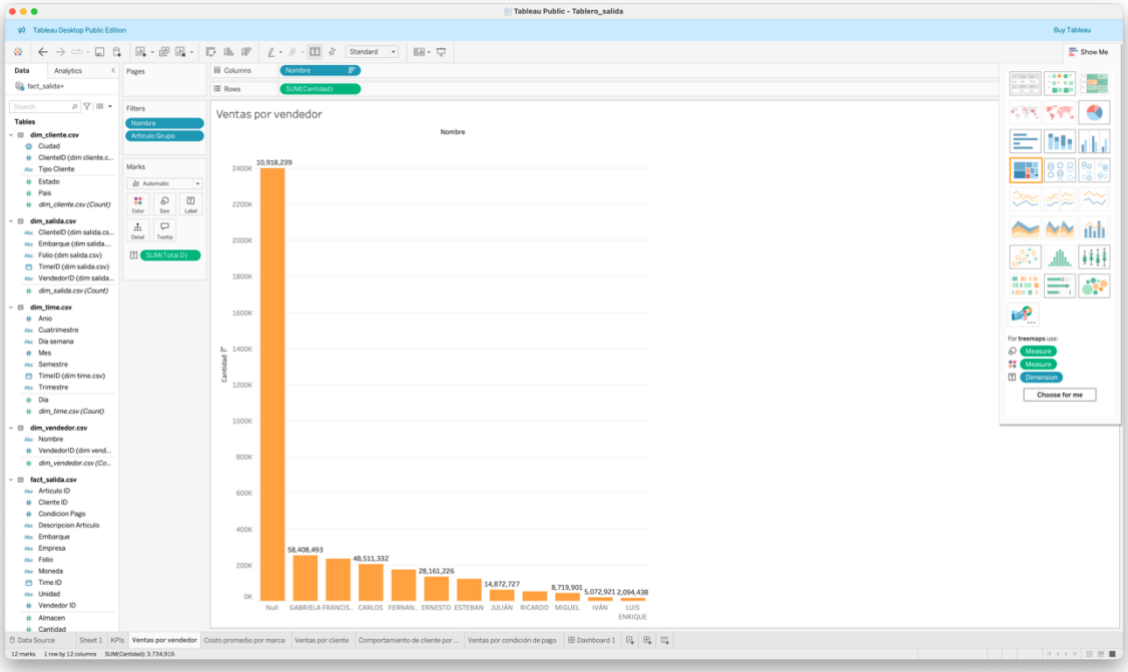


Tableau Public - Tablero_salida
Ventas por vendedor
Nombre
10,918,239
58,408,493
48,511,332
28,161,226
14,872,727
8,719,901
5,072,921
2,094,438
Null
GABRIELA
FRANCIS..
CARLOS
FERNAN..
ERNESTO
ESTEBAN
JULIÁN
RICARDO
MIGUEL
IVÁN
LUIS ENRIQUE

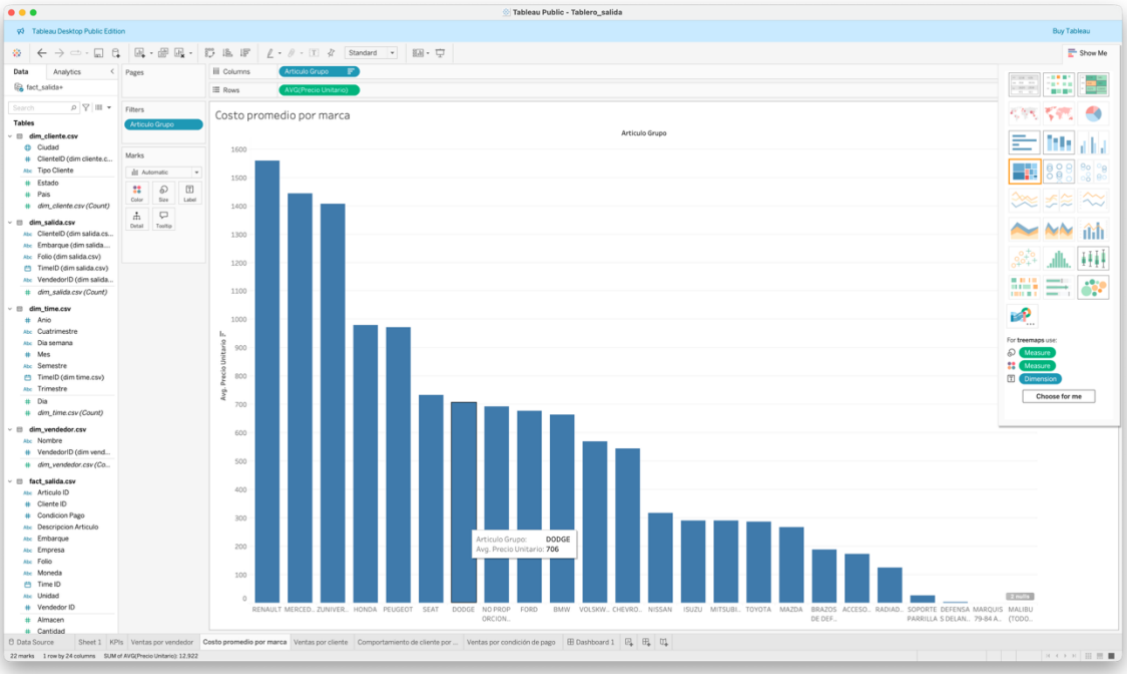


Tableau Public - Tablero_salida
Tableau Desktop Public Edition
Buy Tableau
Show Me
Data
Analytics
fact_salida+
Pages
Filters
Articulo Grupo
Marks
Automatic
Columns
Articulo Grupo
Rows
AVG(Precio Unitario)
Costo promedio por marca
Articulo Grupo
Avg. Precio Unitario
1600
1500
1400
1300
1200
1100
1000
900
800
700
600
500
400
300
200
100
0
Articulo Grupo: DODGE
Avg. Precio Unitario: 706
RENAULT
MERCED..
ZUNIVER..
HONDA
PEUGEOT
SEAT
DODGE
NO PROP ORCION..
FORD
BMW
VOLSKW..
CHEVRO..
NISSAN
ISUZU
MITSUBI..
TOYOTA
MAZDA
BRAZOS DE DEF..
ACCESO..
RADIAD..
SOPORTE PARRILLA S
DEFENSA DELAN..
MARQUIS 79-84 A..
MALIBU (TODO..
For treemaps use:
Measure
Measure
Dimension
Choose for me
Data Source
Sheet 1
KPIs
Ventas por vendedor
Costo promedio por marca
Ventas por cliente
Comportamiento de cliente por ...
Ventas por condición de pago
Dashboard 1
22 marks
1 row by 24 columns
SUM of AVG(Precio Unitario): 12.922

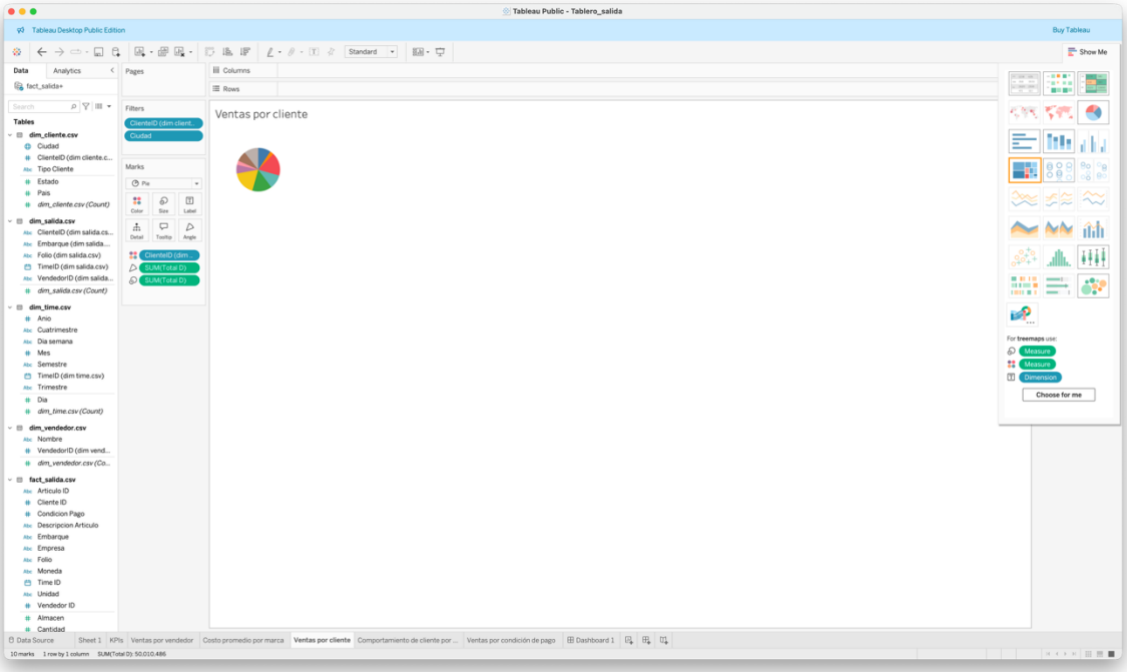


Tableau Public - Tablero_salida
Tableau Desktop Public Edition
Buy Tableau
Show Me
Data
Analytics
fact_salida+
Pages
Filters
ClienteID (dim client..
Ciudad
Marks
Pie
Columns
Rows
Ventas por cliente
For treemaps use:
Measure
Measure
Dimension
Choose for me
Data Source
Sheet 1
KPIs
Ventas por vendedor
Costo promedio por marca
Ventas por cliente
Comportamiento de cliente por ...
Ventas por condición de pago
Dashboard 1
10 marks
1 row by 1 column
SUM(Total D): 50,010,486

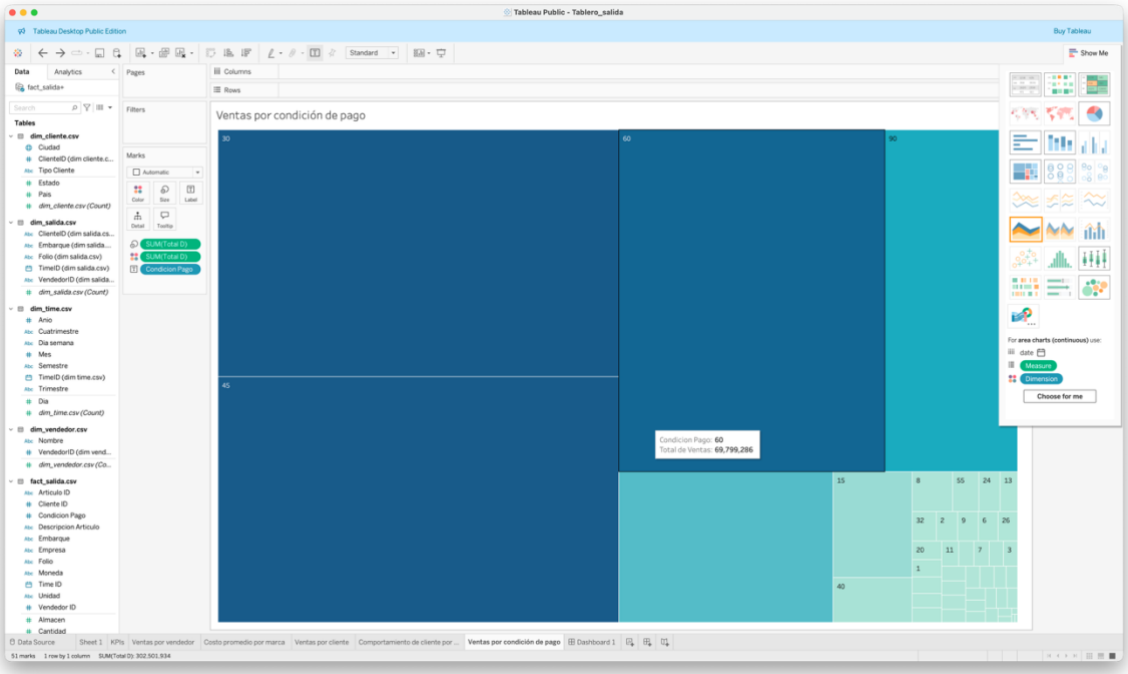


Tableau Public - Tablero_salida
Tableau Desktop Public Edition
Buy Tableau
Ventas por condición de pago
30
60
90
45
15
40
Condicion Pago: 60
Total de Ventas: 69,799,286
Tables
dim_cliente.csv
dim_salida.csv
dim_time.csv
dim_vendedor.csv
fact_salida.csv
Sheet 1
KPIs
Ventas por vendedor
Costo promedio por marca
Ventas por cliente
Comportamiento de cliente por ...
Ventas por condición de pago
Dashboard 1

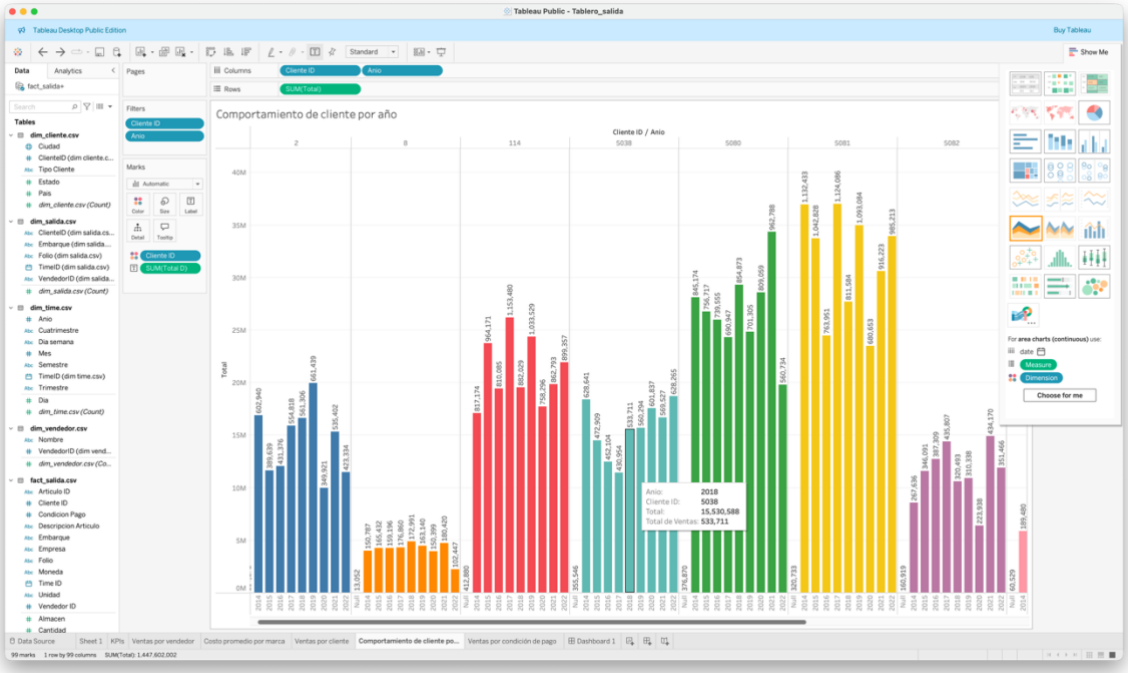
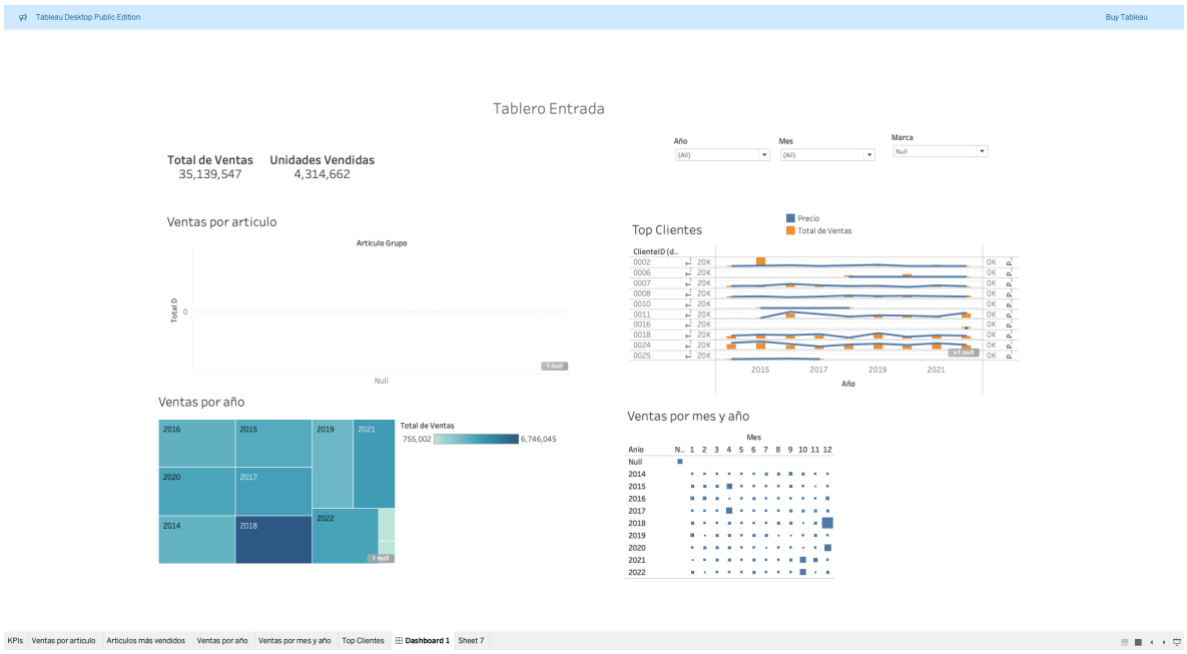


Tableau Public - Tablero_salida
Tableau Desktop Public Edition
Buy Tableau
Comportamiento de cliente por año
Cliente ID / Anio
Total
Anio: 2018
Cliente ID: 5038
Total: 15,530,588
Total de Ventas: 533,711
Tables
dim_cliente.csv
dim_salida.csv
dim_time.csv
dim_vendedor.csv
fact_salida.csv
Sheet 1
KPIs
Ventas por vendedor
Costo promedio por marca
Ventas por cliente
Comportamiento de cliente po...
Ventas por condición de pago
Dashboard 1

Al final, las hojas se integran a un tablero, formando nuestro visual final.

Tablero de entradas



Tablero de salida

III. Análisis de áreas

Análisis de entradas

- **Los diez principales clientes concentran la mayoría del volumen de entradas, eso significa que como empresa se puede tener una dependencia a esos clientes y perder a uno de esos puede representar un riesgo**
- **Los diez artículos más relevantes concentran la mayoría del volumen de entradas, por lo que debería darse prioridad al stock de esos productos**
- **Los tres mejores vendedores abarcan la mayoría de las ventas, esto puede presentar un riesgo en caso de rotación del personal.**
- **Muchos meses tienen picos, no existe un flujo constante por lo que pueden desarrollarse planes de stock por temporadas y/o predecir la demanda por meses**
- **Las ganancias por trimestre son muy similares a excepción del último trimestre. Se pueden definir nuevas estrategias comerciales como descuentos para compensar la baja venta.**
- **Los tickets promedio de cada cliente son extremadamente diferentes. Esto puede permitir a la empresa segmentar a sus clientes y nuevamente hacer estrategias comerciales personalizadas**
- **El cliente número uno representa un volumen gigante, esto puede ser riesgoso en caso de que se pierda el cliente.**
- **Los vendedores tienen ventas muy similares a excepción de dos que pueden considerarse outliers. La empresa puede definir coaching 1:1 con estos vendedores que salen del rango normal.**
- **Hay varios clientes que se podían considerar como clientes frecuentes pero dejaron de comprar. Se pueden definir estrategias de recompra o hacer un acercamiento con ellos para saber que salió mal.**

- **Ya que los 10 principales productos representan gran parte del volumen, se puede hacer una alianza comercial con el proveedor para reducir el precio a cambio de más volumen de entradas.**
- **Hay prodctos con ventas mínimas, la empresa puede eliminarlo de las entradas o de compras frecuentes para incrementar el costo al cliente final y redicir espacio en inventario.**
- **Hay muy pocos datos de clientes nuevos, se puede definir una estrategia comercial para atrer a más clientes nuevos.**
- **Hay muchos clientes que compran mucho en un solo mes y no vuelven a comprar, se pueden definir estrategias comerciales como un programa de fidelidad para fomentar la recompra.**
- **Muchas de las entradas son de productos relacionados, pueden venderse en paquete o hacer promociones para aumentar el ticket promedio.**
- **Hay una ciudad que es prácticamente el 90% del volumen de ventas, muchas varias ciudades con pocas ventas y ciudades con 0 ventas. Quizá valga la pena atacar ciudad por ciudad o concentrar todo el esfuerzo y recursos en la ciudad con el mayor porcentaje de ventas.**

Análisis de Salida

- **Los descuentos varían mucho por vendedor, quizá valga la pena definir topes para garantizar que los vendedores no regalen dinero.**
- **Se venden muchos productos con bajo costo, se pueden hacer paquetes para ganar más por salida.**
- **Los productos con precios intermedios se venden menos.**
- **Los tiempos entre compras varían mucho por cliente, se pueden hacer estrategias comerciales para incrementar la frecuencia de venta**
- **Ciertos clientes representan un gran porcentaje de salidas, aquí también es riesgosa la dependencia a un solo cliente o grupo de clientes**
- **Los artículos por grupo tienen picos, pueden definirse campañas par cuando hay picos y cuando la venta es regular.**

- **Aunque los productos de renault son los más caros, representan un porcentaje muy bajo de las ventas**
- **Aunque no es una diferencia muy grande, hay condiciones de pago muy poco utilizadas, se pueden hacer estrategias para premiar las más usadas o las que más convenga a la empresa y/o estrategias de uso de condicion de pago por segmento de cliente**
- **Hay meses con mucho descuento y ventas no tan buenas, lo que resulta en mucho menos ingreso para la empresa.**
- **Los articulos más vendidos son productos similares pero de diferentes marcas, aumentar el catálogo con productos similares puede ayudar a incrementar las ventas**
- **Muchos clientes compran siempre el mismo producto, se pueden ofrecer productos complentarios para incrementar el ticket promedio**
- **Algunos productos se venden bien aunque no tengan descuento**
- **Clientes nuevos hacen pedidos pequeños, se puede ofrecer un paquete de bienvenida en búsqueda de la recompra**
- **Varios clientes hacen compras frecuentes y seriadas, se puede anticipar la venta y garantizar que compren a nosotros y no a otras empresas**
- **Hay demasiados productos parecidos, se puede agregar descuento a los productos más comprados de cada categoría para incrementar volumen de ventas**

IV. Conclusiones

Alan Daniel: Creo que lo más complicado o lo que le puede dar más valor a un proyecto como este es siempre tener el midset o punto de vista de producto con respecto a las áreas que van a utilizar esta información. La forma en como nos planteamos las necesidades del negocio o las necesidades de información definen el cómo vamos a modelar. El resto creo que es algo sencillo y fácil de dominar con la práctica.

Paulina: Durante el desarrollo del proyecto pude poner en práctica todo lo que he aprendido en el curso, desde la clasificación y creación de nuevas tablas con su respectivo poblado, dentro de un modelo de constelación, hasta la lectura y búsqueda de información dentro de tableros. He podido aprender a corregir errores a la hora de relacionar tablas y al ejecutar el poblado de datos. Asimismo, pude repasar conocimientos prácticos, como la creación de un poblado de datos en KNIME o la construcción de cubos OLAP dentro de Visual Studio.

David: La forma en que se construye el modelo, influye mucho como todo lo demás que se crea , así como también si el modelo es muy complicado afecta directamente en como todo lo demás se debe de hacer, es una buena forma de hacer que la información quede de manera estructurada, y que tableue muy útil para mostrar la información interesante del proyecto.

V. Referencias

- Jonker, A., Holdsworth, J., Kosinski, M. (). *¿Qué es un data warehouse?* IBM. https://www.ibm.com/mx-es/think/topics/data-warehouse
- LinkedIn. (2023, 29 de marzo). *¿Cuáles son los beneficios y desventajas de usar constelación de hechos sobre esquema estelar?* LinkedIn. https://www.linkedin.com/advice/0/what-benefits-drawbacks-using-fact-constellation?lang=es&lang=es&originalSubdomain=es
- Davles, G. (2024, 28 de octubre). *¿Qué es KNIME? Guía introductoria*. Datacamp. https://www.datacamp.com/es/blog/what-is-knime
- Microsoft. (2024, 01 de noviembre). *Introducción a los cubos OLAP de Service Manager para el análisis avanzado.* Microsoft. https://learn.microsoft.com/es-es/system-center/scsm/olap-cubes-overview?view=sc-sm-2025
- Intelisis. (2025, 29 de abril). Transforma la toma de decisiones con tableros de análisis de datos. Intelisis. https://www.intelisis.com/blog/analisis-de-datos/#:~:text=Los%20tableros%20de%20an%C3%A1lisis%20de%20datos%20son%20herramientas%20visuales%20que,y%20niveles%20de%20acceso%20seguros.

VI. Anexos

1. Stored Procedures (SP)

A. Tabla dim_articulo

```
-- Tabla con todos los articulos

CREATE PROCEDURE dbo.sp_dim_articulo
AS
BEGIN
    SELECT
        UPPER(TRIM(a.[clave])) AS ArticuloID,
        UPPER(TRIM(a.[Descripcion])) AS Descripcion,
        UPPER(TRIM(ISNULL(artt.[Descripcion], 'No proporcionado'))) AS
Articulo_Tipo,
        UPPER(TRIM(ISNULL(ag.[Descripcion], 'No proporcionado'))) AS
Articulo_Grupo,
        UPPER(TRIM(ISNULL(ac.[Descripcion], 'No proporcionado'))) AS
Articulo_Clase,
        a.[Precio]

    FROM [AutopartesO2025].[dbo].[Articulo] a

    LEFT JOIN [AutopartesO2025].[dbo].[ArticuloTipo] artt
        ON a.ArticuloTipo = artt.Clave

    LEFT JOIN [AutopartesO2025].[dbo].[ArticuloGrupo] ag
        ON a.ArticuloGrupo = ag.Clave

    LEFT JOIN [AutopartesO2025].[dbo].[ArticuloClase] ac
        ON a.ArticuloClase = ac.Clave
END;
```

B. Tabla dim_cliente

```
-- Tabla con todos los clientes y su tipo

CREATE PROCEDURE dbo.sp_dim_cliente
AS
BEGIN
    SELECT
```

```
        UPPER(TRIM(c.[Clave])) as ClienteID,
        -- Valores bloqueados por privacidad
        UPPER(TRIM(ISNULL(c.[Ciudad], 'No proporcionado'))) AS Ciudad,
        UPPER(TRIM(ISNULL(c.[Estado], 'No proporcionado'))) AS Estado,
        UPPER(TRIM(ISNULL(c.[Pais], 'No proporcionado'))) AS Pais,
        UPPER(TRIM(ISNULL(ct.[Descripcion], 'No proporcionado'))) as
Tipo_Cliente

    FROM [AutopartesO2025].[dbo].[Cliente] c

    LEFT JOIN [AutopartesO2025].[dbo].[ClienteTipo] ct
        ON c.ClienteTipo = ct.Clave
END;
```

C. Tabla dim_entrada

```
-- Tabla con todos los IDs

CREATE PROCEDURE sp_dim_entrada
AS
BEGIN
    SELECT DISTINCT
        UPPER(TRIM(e.[Folio])) AS Folio,
        UPPER(TRIM(ISNULL(e.[Cliente], 'No proporcionado'))) AS
ClienteID,
        UPPER(TRIM(ISNULL(e.[Vendedor], 'No proporcionado'))) AS
VendedorID,
        CAST(CONVERT(varchar(8), Fecha, 112) AS INT) AS TimeID

    FROM [AutopartesO2025].[dbo].[EntradaEncabezado] e
END;
```

D. Tabla dim_salida

```
-- Tabla de IDs de salida

CREATE PROCEDURE sp_dim_salida
AS
BEGIN
    SELECT DISTINCT
        UPPER(TRIM(e.[Folio])) AS Folio,
        UPPER(TRIM(ISNULL(e.[Cliente], 'No proporcionado'))) AS
ClienteID,
```

```
        UPPER(TRIM(ISNULL(e.[Vendedor], 'No proporcionado'))) AS
VendedorID,
        UPPER(TRIM(ISNULL(e.[MedioEmbarque], 'No proporcionado'))) AS
Embarque,
        CAST(CONVERT(varchar(8), Fecha, 112) AS INT) AS TimeID

    FROM [AutopartesO2025].[dbo].[SalidaEncabezado] e
END;
```

E. Tabla dim_time

```
-- Tabla de tiempo que une los tiempos de entrada y salida de c/orden

CREATE PROCEDURE sp_dim_time
AS
BEGIN
    SELECT
        DISTINCT
        -- Convierte la fecha (valor Date) en Int
        CAST(CONVERT(varchar(8), Fecha, 112) AS INT) as TimeID,

        -- Año
        YEAR(oe.Fecha) as Anio,

        -- Semestre (Caso 1 = Ene-Jun, Caso 2 = Jul-Dic)
        CASE
            WHEN MONTH(oe.Fecha) BETWEEN 1 AND 6 THEN '1S'
            ELSE '2S'
        END AS Semestre,

        -- Cuatrimestre (Caso 1 = Ene-Abr, Caso 2 = Mayo-Ago, Caso 3 =
Sept-Dic)
        CASE
            WHEN MONTH(oe.Fecha) BETWEEN 5 AND 8 THEN '2C'
            WHEN MONTH(oe.Fecha) BETWEEN 1 AND 4 THEN '1C'
            ELSE '3C'
        END AS Cuatrimestre,

        -- Trimestre (Caso 1 = Ene-Mar, Caso 2 = Abr-Jun, Caso 3 = Jul-
Sept, Caso 4 = Oct-Dic)
        CONCAT(DATEPART(QUARTER, oe.Fecha), 'T') as Trimestre,

        -- Mes
        MONTH(oe.Fecha) as Mes,
```

```
        -- Dia de la semana
        DATENAME(WEEKDAY, oe.Fecha) AS Dia_semana,

        -- Dia
        DAY(oe.Fecha) as Dia

    FROM [AutopartesO2025].[dbo].[OrdEntradaEncabezado] oe

    UNION

    SELECT
        DISTINCT
        -- Convierte la fecha (valor Date) en Int
        CAST(CONVERT(varchar(8), Fecha, 112) AS INT) as TimeID,

        -- Año
        YEAR(se.Fecha) as Anio,

        -- Semestre (Caso 1 = Ene-Jun, Caso 2 = Jul-Dic)
        CASE
            WHEN MONTH(se.Fecha) BETWEEN 1 AND 6 THEN '1S'
            ELSE '2S'
        END AS Semestre,

        -- Cuatrimestre (Caso 1 = Ene-Abr, Caso 2 = Mayo-Ago, Caso 3 =
Sept-Dic)
        CASE
            WHEN MONTH(se.Fecha) BETWEEN 5 AND 8 THEN '2C'
            WHEN MONTH(se.Fecha) BETWEEN 1 AND 4 THEN '1C'
            ELSE '3C'
        END AS Cuatrimestre,

        -- Trimestre (Caso 1 = Ene-Mar, Caso 2 = Abr-Jun, Caso 3 = Jul-
Sept, Caso 4 = Oct-Dic)
        CONCAT(DATEPART(QUARTER, se.Fecha), 'T') as Trimestre,

        -- Mes
        MONTH(se.Fecha) as Mes,

        -- Dia de la semana
        DATENAME(WEEKDAY, se.Fecha) AS Dia_semana,

        -- Dia
        DAY(se.Fecha) as Dia
```

```
    FROM [AutopartesO2025].[dbo].[OrdSalidaEncabezado] se
END;
```

F. Tabla dim_vendedor

```
-- Tabla con todos los vendedores

CREATE PROCEDURE sp_dim_vendedor
AS
BEGIN
    SELECT DISTINCT
        UPPER(TRIM(v.[Clave])) as VendedorID,
        UPPER(TRIM(v.[Nombre])) AS Nombre

    FROM [AutopartesO2025].[dbo].[Vendedor] v
END;
```

G. Tabla fact_entrada

```
-- Tabla fact principal, registra las entradas

CREATE PROCEDURE [dbo].[sp_fact_entrada]
AS
BEGIN
    SELECT
        UPPER(TRIM(e.[Folio])) AS Folio,
        UPPER(TRIM(ISNULL(e.[Cliente], 'No proporcionado'))) AS
ClienteID,
        -- En caso de que VendedorID sea NULL o esta vacio, se escribe NO
PROPORCIONADO
        CASE
            WHEN UPPER(TRIM(ISNULL(e.[Vendedor], ''))) = '' THEN 'NO
PROPORCIONADO'
            ELSE UPPER(TRIM(e.[Vendedor]))
        END AS VendedorID,
        UPPER(TRIM(d.[Articulo])) AS ArticuloID,
        UPPER(TRIM(ISNULL(d.[DescripcionArticulo], 'No proporcionado')))
AS Descripcion_Articulo,
        UPPER(TRIM(ISNULL(d.[CodigoAlternoArticulo], 'No
proporcionado'))) AS Codigo_Articulo,
        CAST(CONVERT(varchar(8), Fecha, 112) AS INT) AS TimeID,
```

```
        UPPER(TRIM(e.[Empresa])) AS Empresa,
        UPPER(TRIM(d.[Almacen])) AS Almacen,
        UPPER(TRIM(d.[Ubicacion])) AS Ubicacion,
        d.[Partida],
        UPPER(TRIM(e.[Operacion])) AS Operacion,

        UPPER(TRIM(e.[Moneda])) AS Moneda,
        d.[Cantidad],
        d.[Precio] AS Precio_Unitario,
        UPPER(TRIM(ISNULL(d.[UMedPartida], 'No proporcionado'))) AS
Unidad,
        d.[CantidadUMedInv] AS Cantidad_Unidad,

        d.[pctDescuento] AS Pct_Descuento,
        ISNULL(e.[pctDescuentoGlobal], 0) AS Pct_Descuento_Global,
        d.[pctImpuesto] AS Pct_Impuesto,

        e.[TotalDescuento] AS Total_Descuento,
        d.[TotalDescuento] AS Total_Descuento_D,

        e.[TotalImporte] AS Total_Importe,
        d.[TotalImporte] AS Total_Importe_D,

        e.[TotalImpuesto] AS Total_Impuesto,
        d.[TotalImpuesto] AS Total_Impuesto_D,

        e.[Total] AS Total,
        d.[Total] AS Total_D

    FROM [AutopartesO2025].[dbo].[EntradaEncabezado] e

    LEFT JOIN [AutopartesO2025].[dbo].[EntradaDetalle] d
        ON e.Folio = d.Folio
END;
```

H. Tabla fact_salida

```
-- Tabla fact principal, registra las salidas

CREATE PROCEDURE sp_fact_salida
AS
BEGIN
    SELECT
        UPPER(TRIM(e.[Folio])) AS Folio,
```

```
        UPPER(TRIM(ISNULL(e.[Cliente], 'No proporcionado'))) AS
ClienteID,
        UPPER(TRIM(ISNULL(e.[Vendedor], 'No proporcionado'))) AS
VendedorID,
        UPPER(TRIM(d.[Articulo])) AS ArticuloID,
        UPPER(TRIM(ISNULL(d.[DescripcionArticulo], 'No proporcionado')))
AS Descripcion_Articulo,
        UPPER(TRIM(ISNULL(e.[MedioEmbarque], 'No proporcionado'))) AS
Embarque,
        CAST(CONVERT(varchar(8), Fecha, 112) AS INT) AS TimeID,

        UPPER(TRIM(e.[Empresa])) AS Empresa,
        UPPER(TRIM(d.[Almacen])) AS Almacen,
        UPPER(TRIM(d.[Ubicacion])) AS Ubicacion,
        d.[Partida],

        UPPER(TRIM(ISNULL(e.[CondicionPago], 'No proporcionado'))) AS
Condicion_Pago,
        UPPER(TRIM(e.[Moneda])) AS Moneda,
        d.[Cantidad],
        d.[Precio] AS Precio_Unitario,
        UPPER(TRIM(d.[UMedPartida])) AS Unidad,
        d.[CantidadUMedInv] AS Cantidad_Unidad,

        d.[pctDescuento] AS Pct_Descuento,
        ISNULL(e.[pctDescuentoGlobal], 0) AS Pct_Descuento_Global,
        d.[pctImpuesto] AS Pct_Impuesto,

        e.[TotalDescuento] AS Total_Descuento,
        d.[TotalDescuento] AS Total_Descuento_D,

        e.[TotalImporte] AS Total_Importe,
        d.[TotalImporte] AS Total_Importe_D,

        e.[TotalImpuesto] AS Total_Impuesto,
        d.[TotalImpuesto] AS Total_Impuesto_D,

        e.[Total] AS Total,
        d.[Total] AS Total_D

    FROM [AutopartesO2025].[dbo].[SalidaEncabezado] e

    LEFT JOIN [AutopartesO2025].[dbo].[SalidaDetalle] d
        ON e.Folio = d.Folio
END;
```

2. Creación de cubos

Con el fin de hacer dos cubos OLAP: uno de *Entradas* y otro de *Salidas*, se utilizan los siguientes códigos MDX para su respectiva creación.

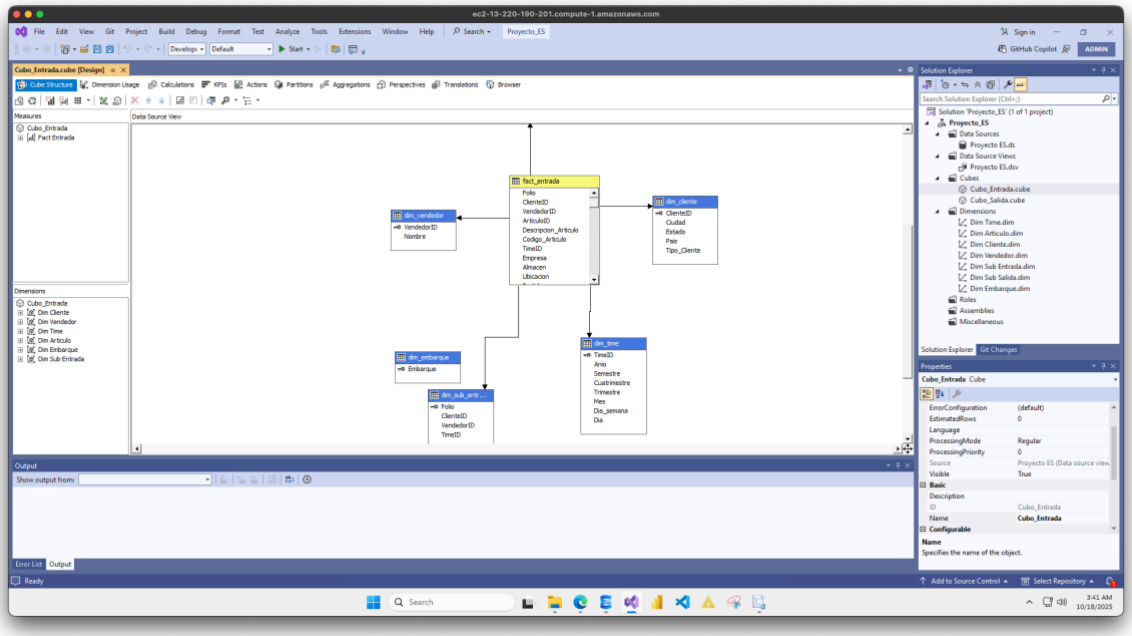
```
CREATE GLOBAL CUBE [Cubo_Entrada_Offline]
STORAGE 'C:\Cubo_Entrada.cub'
FROM [Cubo_Entrada]
(
  MEASURE [Cubo_Entrada].[Cantidad],
  MEASURE [Cubo_Entrada].[Cantidad Unidad],
  MEASURE [Cubo_Entrada].[Fact Entrada Count],
  MEASURE [Cubo_Entrada].[Partida],
  MEASURE [Cubo_Entrada].[Pct Descuento],
  MEASURE [Cubo_Entrada].[Pct Descuento Global],
  MEASURE [Cubo_Entrada].[Pct Impuesto],
  MEASURE [Cubo_Entrada].[Precio Unitario],
  MEASURE [Cubo_Entrada].[Total],
  MEASURE [Cubo_Entrada].[Total D],
  MEASURE [Cubo_Entrada].[Total Descuento],
  MEASURE [Cubo_Entrada].[Total Descuento D],
  MEASURE [Cubo_Entrada].[Total Importe],
  MEASURE [Cubo_Entrada].[Total Importe D],
  MEASURE [Cubo_Entrada].[Total Impuesto],
  MEASURE [Cubo_Entrada].[Total Impuesto D],

  DIMENSION [Cubo_Entrada].[Dim Articulo],
  DIMENSION [Cubo_Entrada].[Dim Cliente],
  DIMENSION [Cubo_Entrada].[Dim Embarque],
  DIMENSION [Cubo_Entrada].[Dim Sub Entrada],
  DIMENSION [Cubo_Entrada].[Dim Time],
  DIMENSION [Cubo_Entrada].[Dim Vendedor]
);
```

```
CREATE GLOBAL CUBE [Cubo_Salida_Offline]
STORAGE 'C:\Cubo_Salida.cub'
FROM [Cubo_Salida]
(
  MEASURE [Cubo_Salida].[Cantidad],
  MEASURE [Cubo_Salida].[Cantidad Unidad],
  MEASURE [Cubo_Salida].[Fact Salida Count],
  MEASURE [Cubo_Salida].[Partida],
  MEASURE [Cubo_Salida].[Pct Descuento],
  MEASURE [Cubo_Salida].[Pct Descuento Global],
  MEASURE [Cubo_Salida].[Pct Impuesto],
  MEASURE [Cubo_Salida].[Precio Unitario],
  MEASURE [Cubo_Salida].[Total],
  MEASURE [Cubo_Salida].[Total D],
  MEASURE [Cubo_Salida].[Total Descuento],
  MEASURE [Cubo_Salida].[Total Descuento D],
  MEASURE [Cubo_Salida].[Total Importe],
  MEASURE [Cubo_Salida].[Total Importe D],
  MEASURE [Cubo_Salida].[Total Impuesto],
  MEASURE [Cubo_Salida].[Total Impuesto D],

  DIMENSION [Cubo_Salida].[Dim Articulo],
  DIMENSION [Cubo_Salida].[Dim Cliente],
  DIMENSION [Cubo_Salida].[Dim Embarque],
  DIMENSION [Cubo_Salida].[Dim Sub Salida],
  DIMENSION [Cubo_Salida].[Dim Time],
  DIMENSION [Cubo_Salida].[Dim Vendedor]
);
```

3. Visual Studio


ec2-13-220-190-201.compute-1.amazonaws.com
Cubo_Entrada.cube [Design]
Cube Structure
Dimension Usage
Calculations
KPIs
Actions
Partitions
Aggregations
Perspectives
Translations
Browser
Measures
Cubo_Entrada
Fact Entrada
Data Source View
fact_entrada
Folio
ClienteID
VendedorID
ArticuloID
Descripcion_Articulo
Codigo_Articulo
TimeID
Empresa
Almacen
Ubicacion
dim_vendedor
VendedorID
Nombre
dim_cliente
ClienteID
Ciudad
Estado
Pais
Tipo_Cliente
dim_embarque
Embarque
dim_sub_entr...
Folio
ClienteID
VendedorID
TimeID
dim_time
TimeID
Anio
Semestre
Cuatrimestre
Trimestre
Mes
Dia_semana
Dia
Dimensions
Cubo_Entrada
Dim Cliente
Dim Vendedor
Dim Time
Dim Articulo
Dim Embarque
Dim Sub Entrada
Output
Show output from:
Error List
Output
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'Proyecto_ES' (1 of 1 project)
Proyecto_ES
Data Sources
Proyecto ES.ds
Data Source Views
Proyecto ES.dsv
Cubes
Cubo_Entrada.cube
Cubo_Salida.cube
Dimensions
Dim Time.dim
Dim Articulo.dim
Dim Cliente.dim
Dim Vendedor.dim
Dim Sub Entrada.dim
Dim Sub Salida.dim
Dim Embarque.dim
Roles
Assemblies
Miscellaneous
Solution Explorer
Git Changes
Properties
Cubo_Entrada Cube
ErrorConfiguration (default)
EstimatedRows 0
Language
ProcessingMode Regular
ProcessingPriority 0
Source Proyecto ES (Data source view)
Visible True
Basic
Description
ID Cubo_Entrada
Name Cubo_Entrada
Configurable
Name
Specifies the name of the object.
Ready
Add to Source Control
Select Repository
Search
3:41 AM
10/18/2025