

29 DE NOVIEMBRE DE 2025



"Implementación de Sistema de Data Warehouse de Datos Financieros y de Ventas con Generación de Reportes Interactivos"

PRESENTA: SANDRA NOEMY NAVAS DE ALVARENGA – ND221932

GRUPO 4
INGENIERÍA EN CIENCIAS DE LA COMPUTACIÓN
DOCENTE: KARENS MEDRANO

INDICE

| | |
|--|----|
| 1. DEFINICIÓN DEL PROBLEMA | 2 |
| Contexto y necesidad empresarial..... | 2 |
| Problema específico identificado..... | 2 |
| Impacto del Problema..... | 2 |
| Solución Propuesta mediante Data Warehouse | 2 |
| 2. RECOLECCIÓN Y PREPARACIÓN DE DATOS | 3 |
| Fuentes de datos identificadas | 3 |
| Sincronización de datos y ejecución de ETL..... | 3 |
| Integración del Data Warehouse..... | 9 |
| 3. IMPORTE DE DATOS A POWER BI | 13 |
| 4. DEFINICIÓN Y DESARROLLO DE KPI's..... | 14 |
| KPIs para el Área de Finanzas..... | 14 |
| KPIs para el Área de Ventas..... | 15 |
| Modelado de KPI's y formulas DAX..... | 17 |
| Tablero de Power BI | 20 |
| 5. CRONOGRAMA DE TRABAJO..... | 22 |
| Fase de Implementación (15 días) | 22 |
| 6. RECURSOS TECNOLÓGICOS REQUERIDOS..... | 23 |

1. DEFINICIÓN DEL PROBLEMA

CONTEXTO Y NECESIDAD EMPRESARIAL

En el entorno empresarial actual, las organizaciones se enfrentan al desafío de gestionar grandes volúmenes de datos financieros y de ventas que se generan diariamente. Estos datos, almacenados comúnmente en archivos Excel, contienen información valiosa pero a menudo permanecen subutilizados debido a la falta de sistemas integrados de análisis.

En esta fase del proyecto, expandiremos el contenido propuesto como solución a la problemática presentada, teniendo en cuenta el contexto de lo presentado anteriormente.

Estado Actual vs Estado Deseado

| Aspecto | Estado Actual | Estado Deseado |
|-------------------------------|----------------------|-------------------------------|
| Consolidación de datos | Manual y dispersa | Centralizada y automatizada |
| Análisis de desempeño | Limitado | Multidimensional y predictivo |
| Reportes | Manuales y estáticos | Interactivos y en tiempo real |
| Toma de decisiones | Basada en intuición | Basada en datos confiables |

PROBLEMA ESPECIFICO IDENTIFICADO

La empresa objetivo carece de una plataforma unificada que permita:

- Consolidar datos financieros dispersos en múltiples archivos Excel.
- Analizar tendencias históricas de ventas y finanzas.
- Monitorear indicadores clave de desempeño en tiempo real.
- Generar reportes automatizados para la toma de decisiones estratégicas.

IMPACTO DEL PROBLEMA

- **Ineficiencia operativa:** Tiempo excesivo en consolidación manual de datos
- **Decisiones subóptimas:** Falta de visibilidad integral del desempeño financiero
- **Oportunidades perdidas:** Incapacidad para identificar tendencias y patrones críticos
- **Riesgo financiero:** Detección tardía de anomalías o desviaciones presupuestarias

SOLUCIÓN PROPUESTA MEDIANTE DATA WAREHOUSE

La implementación de un Data Warehouse permitirá:

- Centralizar todos los datos financieros y de ventas en un repositorio único
- Estandarizar y limpiar la información mediante procesos ETL
- Habilitar análisis multidimensional mediante cubos OLAP
- Generar reportes interactivos con visualización avanzada de KPIs
- Facilitar la toma de decisiones basada en datos históricos y predictivos

2. RECOLECCIÓN Y PREPARACIÓN DE DATOS

FUENTES DE DATOS IDENTIFICADAS

Archivo Excel Principal: **Datos_financieros_ventas.xlsx** contiene las siguientes hojas:

1. **Hoja "Transacciones":**

- ID_Transacción, Fecha, ID_Cliente, ID_Producto, Cantidad, Precio_Unitario, Total
- Campo de estado: Activo/Cancelado/Devuelto

2. **Hoja "Clientes":**

- ID_Cliente, Nombre, Segmento (Minorista/Mayorista/Corporativo), Región, Fecha_Registro

3. **Hoja "Productos":**

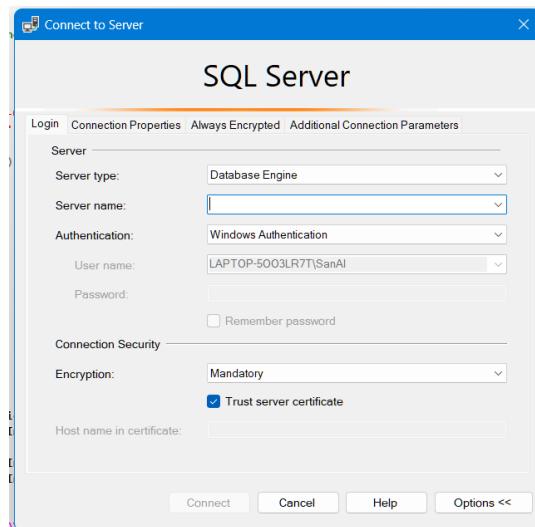
- ID_Producto, Categoría, Subcategoría, Costo_Unitario, Margen_Beneficio

4. **Hoja "Gastos":**

- ID_Gasto, Fecha, Categoría_Gasto (Nómina/Marketing/Operativos), Monto, Departamento

SINCRONIZACIÓN DE DATOS Y EJECUCIÓN DE ETL

Con el apoyo de SQL Server Management Studio (SSMS), generamos nuestra conexión a base de datos local.



Y se programaron una serie de scripts, los cuales cumplen con las siguientes funciones:

a) **Crear la base de datos en caso esta no exista.**

```
/*
=====
1. Crear la base de datos si no existe
===== */
IF NOT EXISTS (SELECT * FROM sys.databases WHERE name = 'DW_FinanzasVentas')
BEGIN
    CREATE DATABASE DW_FinanzasVentas;
END
GO
```

```

USE DW_FinanzasVentas;
GO

/* =====
DIMENSIONES
===== */

/* Dimensión Tiempo */
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID(N'DimTiempo') AND type = 'U')
BEGIN
    CREATE TABLE DimTiempo (
        IdTiempo INT IDENTITY(1,1) PRIMARY KEY,
        Fecha DATE NOT NULL UNIQUE,
        Anio INT NOT NULL,
        Mes INT NOT NULL,
        NombreMes NVARCHAR(20) NOT NULL,
        Trimestre INT NOT NULL,
        Dia INT NOT NULL,
        DiaSemanaNumero INT NOT NULL,
        NombreDiaSemana NVARCHAR(20) NOT NULL
    );
END
GO

/* Dimensión Cliente */
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID(N'DimCliente') AND type = 'U')
BEGIN
    CREATE TABLE DimCliente (
        IdCliente INT NOT NULL PRIMARY KEY,
        Nombre NVARCHAR(150) NOT NULL,
        Segmento NVARCHAR(50) NOT NULL,
        Region NVARCHAR(50) NOT NULL,
        FechaRegistro DATE NULL
    );
END
GO

/* Dimensión Producto */
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID(N'DimProducto') AND type = 'U')
BEGIN
    CREATE TABLE DimProducto (
        IdProducto INT NOT NULL PRIMARY KEY,
        Categoria NVARCHAR(100) NOT NULL,
        Subcategoria NVARCHAR(100) NOT NULL,
        CostoUnitario DECIMAL(18,2) NOT NULL,
        MargenBeneficio DECIMAL(5,2) NULL
    );
END
GO

/* Dimensión Gasto */
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID(N'DimGasto') AND type = 'U')
BEGIN
    CREATE TABLE DimGasto (
        IdGasto INT NOT NULL PRIMARY KEY,
        CategoriaGasto NVARCHAR(100) NOT NULL,

```

```

        Departamento NVARCHAR(100) NOT NULL
    );
END
GO

/*
=====
TABLAS DE HECHOS
===== */
/* Hechos de Ventas */
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID(N'FactVentas')
AND type = 'U')
BEGIN
    CREATE TABLE FactVentas (
        IdVenta BIGINT IDENTITY(1,1) PRIMARY KEY,
        IdTiempo INT NOT NULL,
        IdCliente INT NOT NULL,
        IdProducto INT NOT NULL,
        Cantidad INT NOT NULL,
        IngresoBruto DECIMAL(18,2) NOT NULL,
        CostoTotal DECIMAL(18,2) NOT NULL,
        Utilidad DECIMAL(18,2) NOT NULL,
        Estado NVARCHAR(20) NULL,
        CONSTRAINT FK_FactVentas_DimTiempo FOREIGN KEY (IdTiempo) REFERENCES
DimTiempo(IdTiempo),
        CONSTRAINT FK_FactVentas_DimCliente FOREIGN KEY (IdCliente) REFERENCES
DimCliente(IdCliente),
        CONSTRAINT FK_FactVentas_DimProducto FOREIGN KEY (IdProducto) REFERENCES
DimProducto(IdProducto)
    );
END
GO

/* Hechos de Gastos */
IF NOT EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID(N'FactGastos')
AND type = 'U')
BEGIN
    CREATE TABLE FactGastos (
        IdFactGasto BIGINT IDENTITY(1,1) PRIMARY KEY,
        IdGasto INT NOT NULL,
        IdTiempo INT NOT NULL,
        Monto DECIMAL(18,2) NOT NULL,
        TipoGasto NVARCHAR(100) NULL,
        CONSTRAINT FK_FactGastos_DimGasto FOREIGN KEY (IdGasto) REFERENCES
DimGasto(IdGasto),
        CONSTRAINT FK_FactGastos_DimTiempo FOREIGN KEY (IdTiempo) REFERENCES
DimTiempo(IdTiempo)
    );
END
GO

/*
=====
ÍNDICES PARA RENDIMIENTO
===== */
/* Índices de FactVentas */
IF NOT EXISTS (SELECT * FROM sys.indexes WHERE name = 'IX_FactVentas_IdTiempo' AND
object_id = OBJECT_ID('FactVentas'))
BEGIN
    CREATE NONCLUSTERED INDEX IX_FactVentas_IdTiempo ON FactVentas(IdTiempo);
END

```

```

IF NOT EXISTS (SELECT * FROM sys.indexes WHERE name = 'IX_FactVentas_IdCliente' AND
object_id = OBJECT_ID('FactVentas'))
BEGIN
    CREATE NONCLUSTERED INDEX IX_FactVentas_IdCliente ON FactVentas(IdCliente);
END

IF NOT EXISTS (SELECT * FROM sys.indexes WHERE name = 'IX_FactVentas_IdProducto' AND
object_id = OBJECT_ID('FactVentas'))
BEGIN
    CREATE NONCLUSTERED INDEX IX_FactVentas_IdProducto ON FactVentas(IdProducto);
END
GO

/* Índices de FactGastos */
IF NOT EXISTS (SELECT * FROM sys.indexes WHERE name = 'IX_FactGastos_IdTiempo' AND
object_id = OBJECT_ID('FactGastos'))
BEGIN
    CREATE NONCLUSTERED INDEX IX_FactGastos_IdTiempo ON FactGastos(IdTiempo);
END

IF NOT EXISTS (SELECT * FROM sys.indexes WHERE name = 'IX_FactGastos_IdGasto' AND
object_id = OBJECT_ID('FactGastos'))
BEGIN
    CREATE NONCLUSTERED INDEX IX_FactGastos_IdGasto ON FactGastos(IdGasto);
END
GO

```

b) Generar datos en base de datos creada

Para efectos de la demostración, este script no solo mueve datos y los almacena en el excel de destino, sino que crea una serie de datos de información que nos servira para demostrar la capacidad de la herramienta como solución al problema propuesto.

```

USE DW_FinanzasVentas;
GO

-- Borrar datos existentes de manera segura
DELETE FROM DimTiempo;
GO

DECLARE
    @FechaInicio DATE = '2015-01-01',
    @FechaFin DATE = '2035-12-31';

WHILE (@FechaInicio <= @FechaFin)
BEGIN
    INSERT INTO DimTiempo (
        Fecha,
        Anio,
        Mes,
        NombreMes,
        Trimestre,
        Dia,
        DiaSemanaNumero,
        NombreDiaSemana
    )
    VALUES (
        @FechaInicio,
        YEAR(@FechaInicio),

```

```

MONTH(@FechaInicio),
DATENAME(MONTH, @FechaInicio),
DATEPART(QUARTER, @FechaInicio),
DAY(@FechaInicio),
DATEPART(WEEKDAY, @FechaInicio),
DATENAME(WEEKDAY, @FechaInicio)
);

SET @FechaInicio = DATEADD(DAY, 1, @FechaInicio);
END;

SELECT TOP 50 * FROM DimTiempo;

-- Crear tabla de números (1 a 50,000)
IF OBJECT_ID('tempdb..#Nums') IS NOT NULL DROP TABLE #Nums;

WITH N AS (
    SELECT 1 AS n
    UNION ALL
    SELECT n + 1 FROM N WHERE n < 50000
)
SELECT n INTO #Nums FROM N OPTION (MAXRECURSION 0);

-- Limpiar la tabla
DELETE FROM DimCliente;

INSERT INTO DimCliente (IdCliente, Nombre, Segmento, Region, FechaRegistro)
SELECT TOP 300
n AS IdCliente,
CONCAT('Cliente ', n) AS Nombre,
CASE (n % 3)
    WHEN 0 THEN 'Minorista'
    WHEN 1 THEN 'Mayorista'
    ELSE 'Corporativo'
END AS Segmento,
CASE (n % 4)
    WHEN 0 THEN 'Centro'
    WHEN 1 THEN 'Occidente'
    WHEN 2 THEN 'Oriente'
    ELSE 'Norte'
END AS Region,
DATEADD(DAY, -n, GETDATE()) AS FechaRegistro
FROM #Nums
ORDER BY n;

DELETE FROM DimProducto;

INSERT INTO DimProducto (IdProducto, Categoria, Subcategoria, CostoUnitario,
MargenBeneficio)
SELECT TOP 60
n AS IdProducto,
CASE (n % 4)
    WHEN 0 THEN 'Electrónica'
    WHEN 1 THEN 'Hogar'
    WHEN 2 THEN 'Ropa'
    ELSE 'Oficina'
END AS Categoria,
CASE (n % 4)
    WHEN 0 THEN 'Premium'
    WHEN 1 THEN 'Standard'
    WHEN 2 THEN 'Básico'

```

```

        ELSE 'Económico'
    END AS Subcategoria,
    (n % 200) + 5 AS CostoUnitario,
    (n % 25) + 5 AS MargenBeneficio
FROM #Nums
ORDER BY n;

DELETE FROM DimGasto;

INSERT INTO DimGasto (IdGasto, CategoriaGasto, Departamento)
SELECT TOP 200
    n AS IdGasto,
    CASE (n % 4)
        WHEN 0 THEN ' Nómina'
        WHEN 1 THEN 'Marketing'
        WHEN 2 THEN 'Operativos'
        ELSE 'Logística'
    END AS CategoriaGasto,
    CASE (n % 4)
        WHEN 0 THEN 'Finanzas'
        WHEN 1 THEN 'Ventas'
        WHEN 2 THEN 'RRHH'
        ELSE 'Gerencia'
    END AS Departamento
FROM #Nums
ORDER BY n;

DELETE FROM FactVentas;

INSERT INTO FactVentas (IdTiempo, IdCliente, IdProducto, Cantidad, IngresoBruto,
CostoTotal, Utilidad, Estado)
SELECT TOP 20000
    (SELECT TOP 1 IdTiempo FROM DimTiempo ORDER BY NEWID()) AS IdTiempo,
    (SELECT TOP 1 IdCliente FROM DimCliente ORDER BY NEWID()) AS IdCliente,
    (SELECT TOP 1 IdProducto FROM DimProducto ORDER BY NEWID()) AS IdProducto,
    (n % 10) + 1 AS Cantidad,
    0 AS IngresoBruto,
    0 AS CostoTotal,
    0 AS Utilidad,
    CASE (n % 10)
        WHEN 0 THEN 'Cancelado'
        WHEN 1 THEN 'Devuelto'
        ELSE 'Activo'
    END AS Estado
FROM #Nums
ORDER BY n;

UPDATE F
SET
    IngresoBruto = F.Cantidad * (P.CostoUnitario * (1 + (P.MargenBeneficio / 100.0))),
    CostoTotal = F.Cantidad * P.CostoUnitario,
    Utilidad = (F.Cantidad * (P.CostoUnitario * (1 + (P.MargenBeneficio / 100.0)))) -
    (F.Cantidad * P.CostoUnitario)
FROM FactVentas F
JOIN DimProducto P ON F.IdProducto = P.IdProducto;

DELETE FROM FactGastos;

INSERT INTO FactGastos (IdGasto, IdTiempo, Monto, TipoGasto)
SELECT TOP 3000
    (SELECT TOP 1 IdGasto FROM DimGasto ORDER BY NEWID()) AS IdGasto,

```

```

(SELECT TOP 1 IdTiempo FROM DimTiempo ORDER BY NEWID()) AS IdTiempo,
(n % 5000) + 50 AS Monto,
CategoriaGasto
FROM DimGasto D
JOIN #Nums N ON 1 = 1
ORDER BY N.n;

```

INTEGRACIÓN DEL DATA WAREHOUSE

Proceso de Extracción:

- Conexión directa al archivo Excel mediante conectores ODBC.
- Extracción incremental basada en timestamps para optimizar rendimiento.
- Validación de integridad de datos durante la extracción.

Para lograr el proceso de extracción del servidor de SQL a nuestro excel de manera exitosa, se procedio a utilizar la herramienta de Python.

Antes de comenzar, se prepararon las carpetas y archivos necesarios antes de ejecutar las acciones, quedando de la siguiente manera:

- DW_Project
 - Data
 - Datos_financieros_ventas.xlsx
 - Logs
 - Archivos automaticos creados como respaldo/historico de ejecución del sistema ETL.

config.py

Propósito:

- Centralizar la configuración del proyecto.
- Contiene la información de conexión a SQL Server y la ruta del archivo Excel.

Contenido principal:

```

SQL_SERVER = {
    "driver": "{ODBC Driver 17 for SQL Server}",
    "server": "LAPTOP-5OO3LR7T",
    "database": "DW_FinanzasVentas",
    "trusted_connection": "yes"
}

```

```
EXCEL_PATH = "data/Datos_financieros_ventas.xlsx"
```

Uso en el proyecto:

- Permite que todos los demás scripts importen las configuraciones sin repetir credenciales ni rutas.

etl.py

Propósito:

- Realizar el proceso completo de **ETL** (Extract, Transform, Load) para poblar las tablas del Data Warehouse desde el Excel de origen.

Funciones principales:

- `get_sql_connection()`: Conecta a SQL Server usando pyodbc.
- `clear_tables(conn)`: Limpia las tablas FACT primero y DIM después para evitar conflictos de claves.
- `extract_data()`: Lee las hojas de Excel (Transacciones, Clientes, Productos, Gastos).
- `transform_data(df_trans, df_cli, df_prod, df_gast)`: Calcula IngresoBruto, CostoTotal, Utilidad y limpia los datos.
- `load_dim_cliente(conn, df_cli), load_dim_producto(conn, df_prod), load_fact_ventas(conn, df_trans), load_fact_gastos(conn, df_gast)`: Carga los datos transformados en las tablas del DW.
- `run_etl()`: Orquesta todo el flujo ETL desde la limpieza hasta la carga de hechos y dimensiones.

Características clave:

- Logging de acciones para seguimiento de errores (`logs/etl_log.txt`).
- Procesamiento secuencial que asegura integridad referencial.

etl_incremental.py

Propósito:

- Permitir **ETL incremental**, es decir, actualizar el DW solo con los registros nuevos sin borrar datos existentes.

Funciones principales:

- `get_max_dates(conn)`: Obtiene la última fecha registrada en FactVentas y FactGastos.
- `filter_incremental(df_trans, df_gast, max_fecha_ventas, max_fecha_gastos)`: Filtra solo los registros nuevos posteriores a la última fecha.
- `upsert_dim_cliente(conn, df_cli), upsert_dim_producto(conn, df_prod)`: Actualiza (Type 1) dimensiones existentes o inserta nuevas si no existen.
- `load_fact_ventas_incremental(conn, df_trans_inc), load_fact_gastos_incremental(conn, df_gast_inc)`: Inserta los registros nuevos en las tablas FACT.
- `run_etl_incremental()`: Orquesta todo el flujo incremental, con logging en `logs/etl_incremental_log.txt`.

Características clave:

- Evita duplicados y mantiene el DW actualizado.
- Convierte correctamente las fechas a tipo date para evitar errores en la relación con DimTiempo.

export_to_excel.py

Propósito:

- Generar un **archivo Excel de respaldo o de exportación** con las tablas actuales del DW, listo para uso o validación de datos.

Funciones principales:

- `get_sql_connection()`: Conecta a SQL Server.
- `export_to_excel()`:
 - Extrae los datos de FactVentas, DimCliente, DimProducto y FactGastos mediante consultas SQL.
 - Crea un archivo Excel con hojas separadas por cada tabla (Transacciones, Clientes, Productos, Gastos).
 - Guarda el archivo en la ruta definida en EXCEL_PATH.

Características clave:

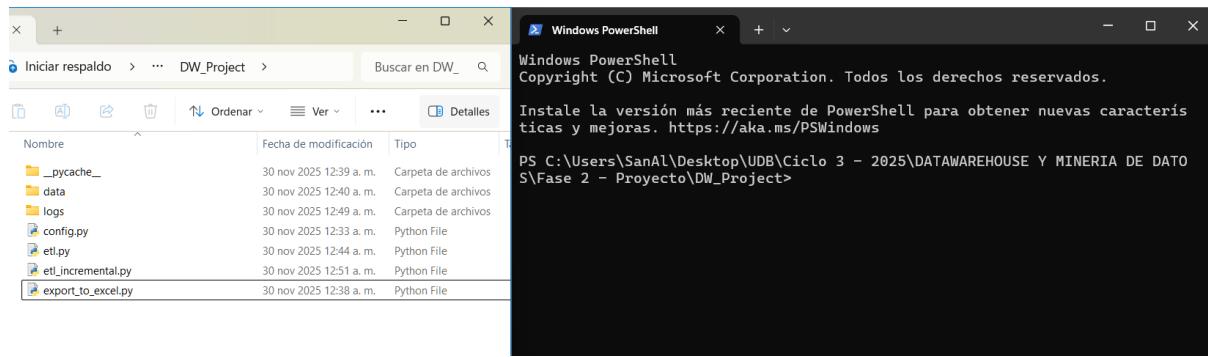
- Facilita la revisión de los datos cargados y sirve como fuente para pruebas o análisis externo.
- Usa `pandas.ExcelWriter` con `openpyxl` para compatibilidad con Excel moderno.

Resumen del flujo con los scripts .py:

| <i>Script</i> | <i>Propósito</i> | <i>Entrada</i> | <i>Salida</i> |
|---------------------------|------------------------------|---|--|
| <i>config.py</i> | Centraliza configuración | N/A | Configuración accesible para todos los scripts |
| <i>etl.py</i> | ETL completo (carga inicial) | Excel con transacciones, clientes, productos y gastos | Tablas DW pobladas (Dim + Fact) |
| <i>etl_incremental.py</i> | ETL incremental | Excel actualizado y DW existente | Solo registros nuevos cargados, dimensiones actualizadas |
| <i>export_to_excel.py</i> | Exportar DW a Excel | Tablas DW | Excel con todas las tablas del DW |

Ejecutar modulos de Phyton.

- Dando clic derecho, en la carpeta del proyecto, ejecutamos la terminal de windows.



- Ingresamos uno por uno los siguientes comandos:
 - py config.py
 - py etl.py
 - py etl_incremental.py
 - py export_to_excel.py

Con esto ya completamos la sincronización de datos al SQL server y realizamos el proceso ETL.

Estructura programada para el Data Warehouse:

Tablas de Dimensiones:

- **Dim_Tiempo** (Fecha, Semana, Mes, Trimestre, Año, Día_Semana)
- **Dim_Cliente** (ID_Cliente, Nombre, Segmento, Región, Fecha_Registro)
- **Dim_Producto** (ID_Producto, Categoría, Subcategoría, Costo_Unitario)
- **Dim_Gasto** (ID_Gasto, Categoría_Gasto, Departamento)

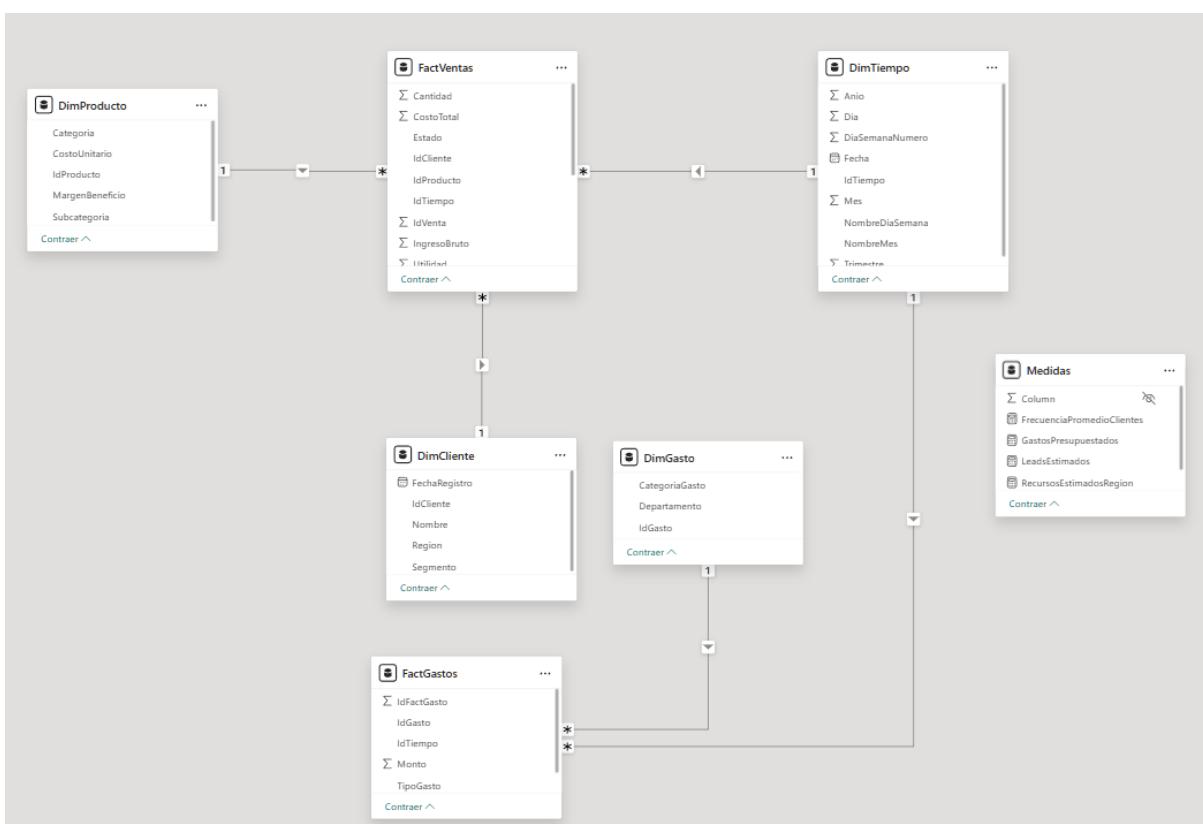
Tablas de Hechos:

- **Fact_Ventas** (ID_Venta, ID_Tiempo, ID_Cliente, ID_Producto, Cantidad, Ingreso_Bruto, Costo_Total, Utilidad)
- **Fact_Gastos** (ID_Gasto, ID_Tiempo, Monto, Tipo_Gasto)

3. IMPORTE DE DATOS A POWER BI

CONEXIÓN Y MODELADO EN POWER BI

- Abrimos Power BI Desktop.
- Seleccionamos **Obtener Datos → SQL Server**.
- Conexión establecida con:
 - Servidor: LAPTOP-5OO3LR7T
 - Base de datos: DW_FinanzasVentas
 - Modo de autenticación: Windows (Trusted Connection)
- Tablas DIM y FACT detectadas por Power BI:
 - DimCliente, DimProducto, DimTiempo, DimGasto
 - FactVentas, FactGastos
- Relaciones entre las tablas usando **modelo en estrella**:
 - FactVentas → DimCliente, DimProducto, DimTiempo
 - FactGastos → DimGasto, DimTiempo



- Se añadio una tabla auxiliar donde alojar las medidas personalizadas obtenidas de entre todos los datos de las diferentes tablas DIM y FACT.
- Se validó que los campos clave coincidieran (ID de dimensiones con FK de hechos).

4. DEFINICIÓN Y DESARROLLO DE KPI'S

KPIS PARA EL ÁREA DE FINANZAS

1. Margen de Beneficio Neto

- **Fórmula:** $\frac{\text{Utilidad Neta}}{\text{Ingresos Totales}} \times 100$
- **Justificación:** Este indicador es el termómetro principal de la salud financiera de la empresa. Un margen decreciente puede indicar problemas de control de costos, presión competitiva en precios, o ineficiencias operativas. Permite a la gerencia tomar decisiones estratégicas sobre precios, reducción de costos y eficiencia operativa.
- **Meta:** Mantener > 18% trimestralmente
- **Frecuencia de monitoreo:** Semanal

2. Flujo de Caja Operativo

- **Fórmula:** Ingresos en Efectivo - Gastos Operativos en Efectivo
- **Justificación:** Mide la capacidad real de generación de efectivo de las operaciones core del negocio. A diferencia de las utilidades contables, el flujo de caja no puede ser manipulado y representa la verdadera liquidez disponible. Es crítico para cumplir con obligaciones a corto plazo y evitar crisis de liquidez.
- **Meta:** Positivo consistentemente con crecimiento del 5% trimestral
- **Frecuencia de monitoreo:** Diaria

3. Rentabilidad por Segmento de Cliente

- **Fórmula:** $\frac{\text{Utilidad por Segmento}}{\text{Ventas por Segmento}} \times 100$
- **Justificación:** Identifica qué tipos de clientes (Minorista/Mayorista/Corporativo) generan mayor rentabilidad. Permite optimizar estrategias comerciales, asignar recursos de atención preferencial a segmentos rentables, y desarrollar programas específicos para mejorar la rentabilidad de segmentos menos rentables.
- **Meta:** Segmento Corporativo > 25% de margen
- **Frecuencia de monitoreo:** Mensual

4. Eficiencia en Control de Gastos

- **Fórmula:** $\frac{\text{Gastos reales}}{\text{Gastos presupuestados}} \times 100$
- **Justificación:** Evalúa la disciplina financiera en la ejecución del presupuesto. Un índice superior al 100% indica sobre-ejecución presupuestaria, mientras que valores muy inferiores pueden señalar sub-ejecución que afecta las operaciones. Fundamental para el control financiero y la rendición de cuentas.
- **Meta:** Mantener entre 95% y 105%
- **Frecuencia de monitoreo:** Mensual

5. Crecimiento Mensual de Utilidad Neta

- **Fórmula:** $\frac{\text{Utilidad Neta Mes Actual} - \text{Utilidad Neta Mes Anterior}}{\text{Utilidad Neta Mes Anterior}} \times 100$
- **Justificación:** Expandida: Mide la trayectoria de crecimiento de la rentabilidad real del negocio. Ayuda a identificar tendencias estacionales, impacto de campañas comerciales, y efectividad de estrategias de reducción de costos. Un crecimiento consistente es indicador de sostenibilidad del modelo de negocio.
- **Meta:** Crecimiento > 8% mensual
- **Frecuencia de monitoreo:** Mensual

KPIS PARA EL ÁREA DE VENTAS

1. Tasa de Conversión por Canal

- **Fórmula:** $\frac{\text{Número de Ventas}}{\text{Número de Leads}} \times 100$
- **Justificación:** Permite identificar qué canales de adquisición de clientes son más efectivos. Una tasa baja puede indicar problemas en el proceso de ventas, calidad de leads, o efectividad del equipo comercial. Facilita la optimización de inversiones en marketing y la reasignación de recursos comerciales.
- **Meta:** > 28% en canal Corporativo
- **Frecuencia de monitoreo:** Mensual

2. Valor de Vida del Cliente (LTV)

- **Fórmula:**
(Ingreso Promedio por Cliente * Frecuencia de Compra) * Vida Promedio del Cliente
- **Justificación:** Fundamental para decisiones de adquisición y retención de clientes. Un LTV alto justifica mayores inversiones en marketing y servicio al cliente. También ayuda a identificar clientes de alto valor que merecen programas de fidelización especiales.
- **Meta:** Incremento del 15% anual
- **Frecuencia de monitoreo:** Trimestral

3. Crecimiento de Ventas Interanual

- **Fórmula:** $\frac{\text{Ventas Período Actual} - \text{Ventas Período Anterior}}{\text{Ventas Período Anterior}} \times 100$
- **Justificación:** Indicador macro del desempeño comercial que elimina efectos estacionales. Un crecimiento positivo sostenido indica salud del mercado y efectividad de estrategias comerciales. Permite comparar desempeño contra benchmarks de la industria.
- **Meta:** > 15% anual
- **Frecuencia de monitoreo:** Mensual

4. Ventas por Segmento de Cliente

- **Fórmula:** $\frac{\text{Ventas Totales por Segmento}}{\text{Ventas Totales}}$
- **Justificación:** Ayuda a entender la composición del portafolio de clientes y dependencia comercial. Una concentración excesiva en un segmento puede representar riesgo estratégico. Guía el desarrollo de estrategias para diversificar la base de clientes.

- **Meta:** Corporativo > 45% del total
- **Frecuencia de monitoreo:** Mensual

5. Eficiencia de Ventas por Región

- **Fórmula:** $\frac{\text{Ventas por Región}}{\text{Recursos Comerciales Asignados}}$
- **Justificación:** Evalúa la productividad comercial por territorio geográfico. Permite identificar regiones sub-atendidas con potencial de crecimiento, optimizar rutas de ventas, y reasignar recursos comerciales hacia áreas de mayor retorno. Esencial para la expansión estratégica.
- **Meta:** Crecimiento del 10% en eficiencia regional
- **Frecuencia de monitoreo:** Trimestral

Tabla Resumen de KPIs

| Área | KPI | Fórmula | Meta | Frecuencia |
|----------|---------------------------|--|------------------|------------|
| Finanzas | Margen de Beneficio Neto | (Utilidad Neta / Ingresos Totales) × 100 | >18% | Semanal |
| Finanzas | Flujo de Caja Operativo | Ingresos - Gastos Operativos | +5% trimestral | Diaría |
| Finanzas | Rentabilidad por Segmento | (Utilidad / Ventas) × 100 | Corporativo >25% | Mensual |
| Finanzas | Control de Gastos | (Gastos reales / Presupuestados) × 100 | 95%-105% | Mensual |
| Finanzas | Crecimiento de Utilidad | ((Mes Actual - Mes Anterior) / Mes Anterior) × 100 | >8% | Mensual |
| Ventas | Conversión por Canal | (Ventas / Leads) × 100 | >28% Corporativo | Mensual |
| Ventas | LTV | Ingreso × Frecuencia × Vida útil | +15% anual | Trimestral |
| Ventas | Crecimiento Interanual | ((Actual - Anterior) / Anterior) × 100 | >15% | Mensual |
| Ventas | Ventas por Segmento | Ventas Segmento / Total | Corporativo >45% | Mensual |
| Ventas | Eficiencia Regional | Ventas / Recursos | +10% | Trimestral |

MODELADO DE KPI'S Y FORMULAS DAX

Se agregaron medidas en Power BI con **DAX**, por ejemplo:

- Margen de Beneficio Neto
- Flujo de Caja Operativo
- Crecimiento de Utilidad
- Ventas por Segmento
- Conversión por Canal
- LTV
- Eficiencia Regional

Para medidas que requerían datos que no existían (ej. Fact_Leads o Fact_Recursos), se **simularon con valores calculados** usando la data existente.

- ✓ Cada medida se agregó en la tabla Fact_Ventas o Medidas según correspondiera.

Resumen de datos por tablas y su tipo en Power BI.

1. DimTiempo

| Campo | Tipo de dato en PBI | Observaciones |
|------------------------|---------------------|---|
| <i>IdTiempo</i> | Entero (Integer) | Clave surrogate, no se muestra al usuario en gráficos |
| <i>Fecha</i> | Fecha (Date) | Usada para ejes de tiempo en gráficos y KPIs |
| <i>Anio</i> | Entero (Integer) | Año calendario |
| <i>Mes</i> | Entero (Integer) | Número de mes |
| <i>NombreMes</i> | Texto (Text) | Nombre del mes (ej. Enero, Febrero) |
| <i>Trimestre</i> | Entero (Integer) | Número de trimestre |
| <i>Dia</i> | Entero (Integer) | Día del mes |
| <i>DiaSemanaNumero</i> | Entero (Integer) | Número de día de la semana (1=Lunes...) |
| <i>NombreDiaSemana</i> | Texto (Text) | Nombre del día de la semana |

2. DimCliente

| Campo | Tipo de dato en PBI | Observaciones |
|----------------------|---------------------|---|
| <i>IdCliente</i> | Entero (Integer) | Clave primaria usada para relaciones con FactVentas |
| <i>Nombre</i> | Texto (Text) | Nombre del cliente |
| <i>Segmento</i> | Texto (Text) | Minorista / Mayorista / Corporativo |
| <i>Region</i> | Texto (Text) | Norte, Sur, Oriente, Occidente |
| <i>FechaRegistro</i> | Fecha (Date) | Fecha de registro del cliente |

3. DimProducto

| Campo | Tipo de dato en PBI | Observaciones |
|------------------------|--------------------------|---|
| <i>IdProducto</i> | Entero (Integer) | Clave primaria usada para relaciones con FactVentas |
| <i>Categoría</i> | Texto (Text) | Categoría del producto |
| <i>Subcategoría</i> | Texto (Text) | Subcategoría del producto |
| <i>CostoUnitario</i> | Decimal (Decimal Number) | Precio base del producto |
| <i>MargenBeneficio</i> | Decimal (Decimal Number) | Porcentaje de margen de beneficio |

4. DimGasto

| Campo | Tipo de dato en PBI | Observaciones |
|-----------------------|---------------------|------------------------------------|
| <i>IdGasto</i> | Entero (Integer) | Clave primaria usada en FactGastos |
| <i>CategoríaGasto</i> | Texto (Text) | Nómina, Marketing, Operativos... |
| <i>Departamento</i> | Texto (Text) | Área responsable del gasto |

5. FactVentas

| Campo | Tipo de dato en PBI | Observaciones |
|---------------------|--------------------------|---|
| <i>IdVenta</i> | Entero (Integer) | Clave surrogate del hecho |
| <i>IdTiempo</i> | Entero (Integer) | FK a DimTiempo, se une para análisis temporal |
| <i>IdCliente</i> | Entero (Integer) | FK a DimCliente |
| <i>IdProducto</i> | Entero (Integer) | FK a DimProducto |
| <i>Cantidad</i> | Entero (Integer) | Cantidad vendida |
| <i>IngresoBruto</i> | Decimal (Decimal Number) | Cantidad * Precio Unitario |
| <i>CostoTotal</i> | Decimal (Decimal Number) | Cantidad * CostoUnitario |
| <i>Utilidad</i> | Decimal (Decimal Number) | IngresoBruto - CostoTotal |
| <i>Estado</i> | Texto (Text) | Activo / Cancelado / Devuelto |

6. FactGastos

| Campo | Tipo de dato en PBI | Observaciones |
|--------------------|--------------------------|----------------------------|
| <i>IdFactGasto</i> | Entero (Integer) | Clave surrogate del hecho |
| <i>IdGasto</i> | Entero (Integer) | FK a DimGasto |
| <i>IdTiempo</i> | Entero (Integer) | FK a DimTiempo |
| <i>Monto</i> | Decimal (Decimal Number) | Monto del gasto |
| <i>TipoGasto</i> | Texto (Text) | Detalle opcional del gasto |

El catálogo de indicadores con sus respectivas fórmulas DAX, se detallan a continuación:

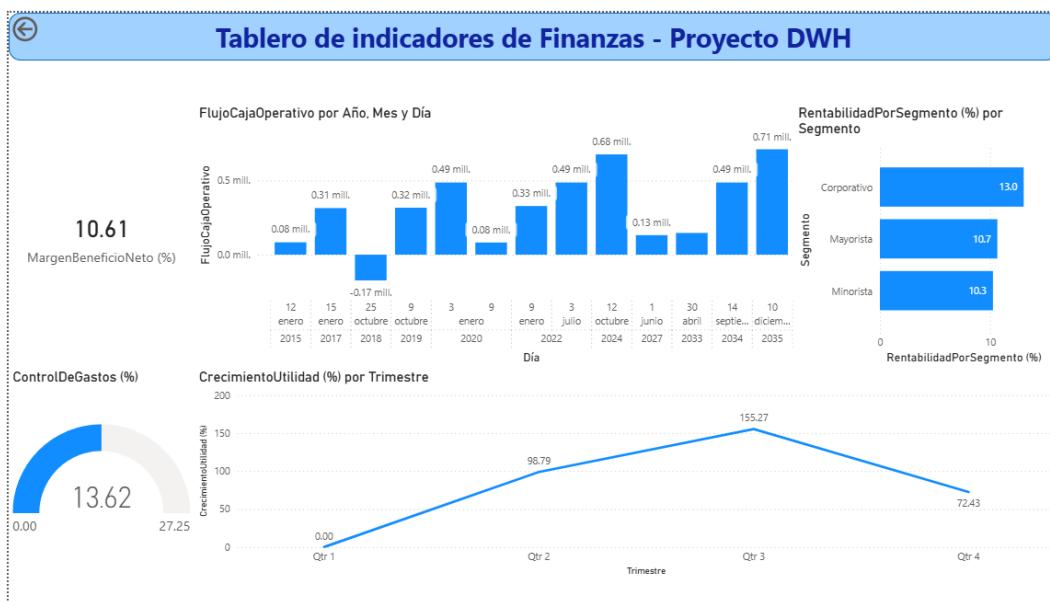
| Área | KPI | Fórmula DAX | Tabla/Medida | Observaciones / Medidas Simuladas |
|----------|---------------------------|---|--------------|---|
| Finanzas | Margen de Beneficio Neto | MargenBeneficioNeto (%) = DIVIDE(SUM(FactVentas[Utilidad]), SUM(FactVentas[IngresoBruto]), 0) * 100 | FactVentas | — |
| Finanzas | Flujo de Caja Operativo | FlujoCajaOperativo = SUM(FactVentas[IngresoBruto]) - SUM(FactGastos[Monto]) | Medidas | — |
| Finanzas | Rentabilidad por Segmento | RentabilidadSegmento (%) = DIVIDE(SUM(FactVentas[Utilidad]), SUM(FactVentas[IngresoBruto]), 0) * 100 | FactVentas | Filtrado por Segmento (ej. Corporativo) |
| Finanzas | Control de Gastos | ControlGastos (%) = DIVIDE(SUM(FactGastos[Monto]), [GastosPresupuestados], 0) * 100 | Medidas | [GastosPresupuestados] = 10000 (simulada por departamento) |
| Finanzas | Crecimiento de Utilidad | CrecimientoUtilidad (%) = DIVIDE(SUM(FactVentas[Utilidad]) - CALCULATE(SUM(FactVentas[Utilidad]), PREVIOUSMONTH(DimTiempo[Fecha])), CALCULATE(SUM(FactVentas[Utilidad]), PREVIOUSMONTH(DimTiempo[Fecha])), 0) * 100 | FactVentas | — |
| Ventas | Conversión por Canal | ConversionPorCanal (%) = DIVIDE(SUM(FactVentas[Cantidad]), [LeadsTotales], 0) * 100 | Medidas | [LeadsTotales] = 250 (simulada por semana/canal) |
| Ventas | LTV | LTV = SUM(FactVentas[IngresoBruto]) * [FrecuenciaPromedioClientes] * [VidaUtilClientes] | Medidas | [FrecuenciaPromedioClientes] = 3 (simulada) [VidaUtilClientes] = 12 meses (simulada) |
| Ventas | Crecimiento Interanual | CrecimientoInteranual (%) = DIVIDE(SUM(FactVentas[IngresoBruto]) - CALCULATE(SUM(FactVentas[IngresoBruto]), SAMEPERIODLASTYEAR(DimTiempo[Fecha])), CALCULATE(SUM(FactVentas[IngresoBruto]), SAMEPERIODLASTYEAR(DimTiempo[Fecha])), 0) * 100 | FactVentas | — |
| Ventas | Ventas por Segmento | VentasPorSegmento (%) = DIVIDE(SUM(FactVentas[IngresoBruto]), CALCULATE(SUM(FactVentas[IngresoBruto]), ALL(DimCliente[Segmento])), 0) * 100 | FactVentas | — |
| Ventas | Eficiencia Regional | EficienciaRegional (%) = DIVIDE(SUM(FactVentas[IngresoBruto]), [RecursosEstimados], 0) * 100 | Medidas | [RecursosEstimados] = 100000 por región (simulada) |

TABLERO DE POWER BI

Generada la conexión de la base de datos, siendo estos los cubos de información, así como las medidas auxiliares y medidas DAX para calcular los indicadores propuestos.

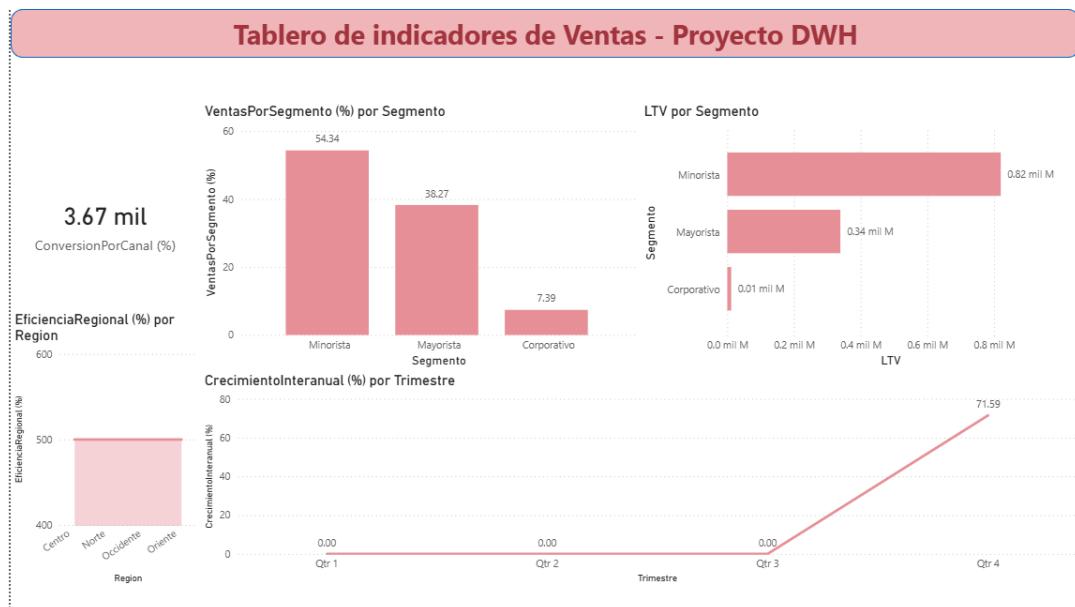
Se diseño un tablero de graficos con las herramientas brindadas por Power BI, siendo su configuración la siguiente:

Hoja 1 - Proyecto DWH – Finanzas



| Visualización | Tipo de gráfico | Datos / Valores | Eje X | Eje Y | Leyenda / Filtros | Observaciones |
|---------------------------|---------------------------------------|---|-----------------|----------------|---------------------------------|--|
| Margen de Beneficio Neto | Tarjeta / KPI | MargenBeneficioNeto (%) | — | — | Filtro periodo | Muestra el margen de utilidad neta. Ideal para control de desempeño financiero. |
| Flujo de Caja Operativo | Gráfico de columnas | FlujoCajaOperativo | Fecha (día/mes) | Monto | Filtro trimestre | Permite ver ingresos menos gastos operativos en la línea de tiempo. |
| Rentabilidad por Segmento | Gráfico de barras agrupadas | RentabilidadSegmento (%) | Segmento | % rentabilidad | Filtro trimestre | Evaluá rentabilidad por cada tipo de cliente. |
| Control de Gastos | Gráfico de columnas con línea de meta | ControlGastos (%) | Departamento | % cumplimiento | Línea de meta 95%-105% | Compara gastos reales contra presupuesto simulado [GastosPresupuestados]. |
| Crecimiento de Utilidad | Gráfico de líneas | CrecimientoUtilidad (%) | Mes | % crecimiento | Filtro por año | Visualiza la evolución mensual de la utilidad neta. |
| Tabla Detalle Finanzas | Tabla / matriz | Monto de FactGastos, IngresoBruto, Utilidad | — | — | Filtro por departamento y fecha | Permite desglosar todas las transacciones financieras y comparar con presupuestos. |

Hoja 2 - Proyecto DWH – Ventas



| Visualización | Tipo de gráfico | Datos / Valores | Eje X | Eje Y | Leyenda / Filtros | Observaciones |
|------------------------|------------------------------------|---|---|-----------------|-------------------------------|--|
| Conversion por Canal | Gráfico de barras agrupadas | ConversionPorCanal (%) | Canal de venta | % de conversión | Filtro por segmento, semana | Muestra la tasa de conversión por canal, útil para comparar efectividad de cada canal de ventas semanal. |
| LTV | Tarjeta / LTV KPI | LTV | — | — | Filtro por Segmento o Cliente | Indica el Valor del Tiempo de Vida de clientes, calculado con medidas simuladas [FrecuenciaPromedioClientes] y [VidaUtilClientes]. |
| Crecimiento Interanual | Gráfico de líneas | CrecimientoInteranual (%) | Año / Trimestre | % crecimiento | Filtro por Segmento | Visualiza la evolución de ventas año con año. Configurar para mostrar todos los trimestres y años. |
| Ventas por Segmento | Gráfico de torta | VentasPorSegmento (%) | Segmento | % ventas | — | Permite ver la participación de cada segmento (Corporativo, Minorista, etc.). |
| Eficiencia Regional | Mapa de coroletas o mapa de formas | EficienciaRegional (%) | Región (Norte, Sur, Oriente, Occidente) | % eficiencia | Filtro por periodo | Simula la eficiencia usando [RecursosEstimados]. Ajustar colores de acuerdo al valor de eficiencia. |
| Tabla Detalle Ventas | Tabla / matriz | ID_Cliente, ID_Producto, Cantidad, IngresoBruto, Utilidad | — | — | Filtro por fechas, segmento | Permite ver detalle transaccional de ventas. |

5. CRONOGRAMA DE TRABAJO

FASE DE IMPLEMENTACIÓN (15 DÍAS)

| Día | Fase / Área | Tarea | Descripción | Responsable |
|-----|-----------------|------------------------------------|---|---------------------------------|
| 1 | Preparación DW | Creación de base de datos y tablas | Crear DW_FinanzasVentas en SQL Server con Dimensiones y Hechos, revisar integridad de claves y tipos de datos. | Sandra Navas – Especialista DWH |
| 2 | Preparación DW | Validación de tablas | Revisar DimTiempo, DimCliente, DimProducto, DimGasto y tablas de hechos, insertar datos de prueba si es necesario. | Sandra Navas – Especialista DWH |
| 3 | ETL Inicial | Configuración scripts | Crear archivos config.py y etl.py, definir rutas de Excel y conexión SQL Server. | Sandra Navas – Especialista DWH |
| 4 | ETL Inicial | Extracción y transformación | Leer Excel, transformar datos (Ingresos, Costos, Utilidad), limpiar datos nulos. | Sandra Navas – Especialista DWH |
| 5 | ETL Inicial | Carga de Dimensiones | Cargar DimCliente y DimProducto en SQL Server. | Sandra Navas – Especialista DWH |
| 6 | ETL Inicial | Carga de Hechos | Cargar FactVentas y FactGastos en SQL Server, validar integridad referencial. | Sandra Navas – Especialista DWH |
| 7 | ETL Incremental | Configuración etl_incremental.py | Adaptar ETL para cargas incrementales, revisar conversión de fechas y filtros por última fecha. | Sandra Navas – Especialista DWH |
| 8 | ETL Incremental | Ejecución y prueba | Ejecutar ETL incremental, validar inserciones, depurar errores de fechas o duplicados. | Sandra Navas – Especialista DWH |
| 9 | Exportación | Configuración export_to_excel.py | Crear script para extraer datos desde SQL Server a Excel para respaldo o validación. | Sandra Navas – Especialista DWH |
| 10 | Power BI | Conexión DW | Conectar Power BI al SQL Server DW, importar todas las tablas (6) y revisar relaciones. | Sandra Navas – Especialista DWH |
| 11 | Power BI | Creación medidas DAX | Programar todas las medidas de KPIs de Finanzas y Ventas, incluyendo simulaciones: [GastosPresupuestados], [LeadsTotales], [FrecuenciaPromedioClientes], [VidaUtilClientes], [RecursosEstimados]. | Sandra Navas – Especialista DWH |
| 12 | Power BI | Validación medidas | Probar cada KPI, revisar resultados, detectar inconsistencias y corregir fórmulas DAX. | Sandra Navas – Especialista DWH |

| | | | | |
|----|-------------------------|-----------------------------|---|---------------------------------|
| 13 | Power BI | Creación de visualizaciones | Seleccionar gráficos adecuados (barras, líneas, torta, mapa), configurar ejes y filtros, incluir metas y comparaciones. | Sandra Navas – Especialista DWH |
| 14 | Power BI | Configuración tablero final | Ordenar visualizaciones, crear panel principal, incluir leyendas, colores, segmentaciones y slicers. | Sandra Navas – Especialista DWH |
| 15 | Documentación y entrega | Generar informe y respaldo | Documentar todo el proceso (ETL, scripts, DAX, visualizaciones), capturas de pantalla, resumen de KPIs, carpeta final lista para entrega. | Sandra Navas – Especialista DWH |

6. RECURSOS TECNOLÓGICOS REQUERIDOS

Infraestructura:

- **Base de Datos:** Microsoft SQL Server 2019+
- **Herramienta ETL:** Python con bibliotecas pandas, pyodbc
- **Reporting:** Power BI Desktop y Service
- **Control de Versiones:** Git/GitHub

Requisitos del Sistema:

- Procesador: 4 cores mínimo
- RAM: 8 GB mínimo (16 GB recomendado)
- Almacenamiento: 50 GB libres
- Sistema Operativo: Windows 10/11 o Windows Server 2019+