* **Nombre del proyecto:** Tienda Aurelion
* **Alumno/a:** [Tu nombre completo]
* **Curso:** [Curso o materia]
* **Docente:** [Nombre del profesor/a]
* **Fecha de entrega:** [Fecha]

### 1.1 Tema, problema y solución

* **Tema:** Inteligencia Artificial para la optimización estratégica de ventas a partir del análisis integrado de datos comerciales en la tienda Aurelion.
* **Problema:** La empresa mantiene un enfoque retrospectivo, sin una visión integradora ni predictiva del negocio. Aunque recopila y dispone de múltiples fuentes de datos valiosos sobre clientes, productos, ventas, detalles de ventas, sucursales, vendedores y métodos de pago, carece de un sistema para analizarlos de forma integrada. Esto conduce a tres problemas centrales:

1. **Desconocimiento del Cliente:** No se identifican patrones de compra, lo que impide reconocer quiénes son los clientes más valiosos o qué productos prefieren, esto limita la capacidad para personalizar acciones comerciales.
2. **Estrategias de Venta Genéricas:**  sin integrar datos de vendedores, descuentos, sucursales o medios de pago, las decisiones se basan en intuición y no en evidencia, resultando en un bajo retorno de inversión y oportunidades de venta perdidas.
3. **Ineficiencia Operativa:** Responder a preguntas de negocio básicas como "¿Cuál fue nuestro producto más vendido el mes pasado?" requieren múltiples pasos manuales y consume mucho tiempo, lo que dificulta la agilidad operativa.

En esencia, la empresa opera de forma reactiva, mirando el pasado sin tener una hoja de ruta clara para tomar decisiones proactivas que impulsen el crecimiento futuro.

* **Solución:** Creación de un Flujo de Trabajo Analítico para Generar Inteligencia de Negocio

La solución consiste en desarrollar un programa en **Python** que automatice el proceso de análisis de datos, *generando inteligencia de negocio a partir de la integración de múltiples fuentes de datos.* transformando la información cruda en conocimiento accionable. Este proceso, delineado en el apartado de pseudocódigo, seguirá estos pasos:

1. **Consolidación de Datos:**

Se unificarán las tablas clientes\_expanded, productos\_expanded, ventas\_expanded, detalle\_ventas\_expanded, sucursales\_expanded, vendedores\_expanded y medios\_pago\_expanded en un único DataFrame maestro para obtener una visión una visión 360° del negocio.

1. **Análisis Descriptivo Automatizado:** El sistema responderá automáticamente a cuestiones clave de negocio, tales como:
   * Identificación de productos y categorías más vendidos.
   * Ranking de los clientes que generan más ingresos.
   * Análisis de ventas según la sucursal, vendedor o medio de pago.
   * Visualización de la evolución de las ventas a lo largo del tiempo.
2. **Fundamento para IA:** En la etapa inicial se sentaron las bases para aplicar técnicas de Inteligencia Artificial (Machine Learning). En esta etapa se establecerá un entorno preparado para aplicar modelos de segmentación de clientes o predicción de demanda en el futuro

Para la implementación, se utilizará **Python,** junto con **Pandas** para manipular datos y **matplotlib - seaborn** para visualización. **GitHub Copilot** actuará como un asistente de codificación para acelerar el desarrollo, sugerir código para la limpieza, unión de datos generación eficiente de código durante el desarrollo de transformaciones, consultas y reportes visuales.

### 1.2 Dataset de referencia

El conjunto de datos para el Proyecto Aurelion consiste en siete tablas interrelacionadas que simulan el entorno transaccional completo de la Tienda Aurelion, con contexto comercial, geográfico y de comportamiento del consumidor.

* **Fuente:** Los 4 archivos .xlsx originales provistos por la cátedra fueron **ampliados y enriquecidos** con tres tablas adicionales (sucursales, vendedores y medios\_pago) para permitir un análisis más granular, conforme a un modelo dimensional tipo estrella, aplicable a entornos reales de inteligencia de negocios.
* **Definición:** Este dataset representa un ecosistema comercial integral, que permite analizar cada venta desde múltiples dimensiones:

| **Pregunta** | **Dimensión asociada** | **Tabla** |
| --- | --- | --- |
| ¿Qué se compra? | Producto, categoría y subcategoría | productos\_expanded |
| ¿Quién compra? | Cliente (nombre, email, ciudad, género, edad) | clientes\_expanded |
| ¿Dónde se compra? | Sucursal específica en Córdoba, Argentina | sucursales\_expanded |
| ¿Quién atiende la venta? | Vendedor asignado a sucursal | vendedores\_expanded |
| ¿Cuándo ocurre la venta? | Fecha de transacción y de alta del cliente | ventas\_expanded, clientes\_expanded |
| ¿Cómo se paga? | Medio de pago estandarizado | medios\_pago\_expanded |
| ¿Cómo se compone la venta? | Encabezado + detalle de ítems | ventas\_expanded + detalle\_ventas\_expanded |

Este modelo permite reconstruir cada transacción y realizar análisis multidimensionales útiles para toma de decisiones estratégicas, como identificar preferencias de clientes, comparar eficiencia por vendedores, medios de pago preferidos y desempeño por sucursal.

* **Escala:** Es un conjunto de datos de muestra, diseñado para el desarrollo y la prueba. Contiene un volumen controlado para facilitar su uso en procesos de exploración, transformación y análisis en el curso:

| **Tabla** | **Número de registros** |
| --- | --- |
| clientes\_expanded | 100 |
| productos\_expanded | 100 |
| ventas\_expanded | 720 |
| detalle\_ventas\_expanded | 2016 |
| sucursales\_expanded | 6 |
| vendedores\_expanded | 36 |
| medios\_pago\_expanded | 4 |

* **Estructura del Modelo:** Los datos están organizados siguiendo un **Modelo Estrella** simplificado, con cinco tablas de **Dimensiones** (CLIENTES, PRODUCTOS, SUCURSALES, VENDEDORES, MEDIOS\_PAGO) que describen las entidades, y dos tablas de **Hechos** (VENTAS, DETALLE\_VENTA) que registran los eventos.

| **Tabla** | **Rol en el Modelo** | **Claves** |
| --- | --- | --- |
| **clientes** | Dimensión | PK: id\_cliente |
| **productos** | Dimensión | PK: id\_producto |
| **sucursales** | Dimensión | PK: id\_sucursal |
| **vendedores** | Dimensión | PK: id\_vendedor, FK: id\_sucursal |
| **medios\_pago** | Dimensión | PK: id\_medio\_pago |
| **ventas** | Hecho (encabezado) | PK: id\_venta, FKs: id\_cliente, id\_sucursal, id\_vendedor, id\_medio\_pago |
| **detalle\_ventas** | Hecho (detalle) | PK compuesto: (id\_venta, id\_producto), FKs: id\_venta, id\_producto |

**Estructura Detallada de las Tablas**

A continuación, se detalla la estructura y el contenido de cada tabla:

**1. Tabla de Dimensión:**CLIENTES\_EXPANDED

* Almacena la información demográfica y de registro de cada cliente.

| **Columna** | **Tipo** | **Descripción** |
| --- | --- | --- |
| id\_cliente | integer (PK) | Identificador único |
| nombre\_cliente | varchar | Nombre completo |
| email | varchar | Correo electrónico |
| ciudad | varchar | Residencia |
| fecha\_alta | date | Alta como cliente |
| genero | varchar | Masculino/Femenino/etc. |
| edad\_rango | varchar | Rango etario (ej. 18–25) |
| activo\_como\_cliente | boolean | Estado activo/inactivo |

**Uso principal:** gestión de información de clientes, trazabilidad histórico-temporal, etc.

**2. Tabla de Dimensión:**PRODUCTOS\_EXPANDED

* Funciona como el catálogo de artículos disponibles para la venta.

| **Columna** | **Tipo** | **Descripción** |
| --- | --- | --- |
| id\_producto | integer (PK) | Identificador del producto |
| nombre\_producto | varchar | Nombre comercial |
| categoria | varchar | Grupo de productos |
| subcategoria | varchar | Subgrupo o línea |
| precio\_unitario | decimal | Precio por unidad |

**Uso principal:** catálogo principal para ventas y análisis de mix de productos.

**3. Tabla de Dimensión:**SUCURSALES\_EXPANDED

* Información geográfica de cada punto de venta.

| **Columna** | **Tipo** | **Descripción** |
| --- | --- | --- |
| id\_sucursal | integer (PK) | ID de la sucursal |
| nombre\_sucursal | varchar | Nombre comercial |
| ciudad | varchar | Ciudad operativa |
| provincia | varchar | Ubicación regional (Córdoba) |

**Uso principal:** segmentación geográfica y operaciones por punto de venta.

**4. Tabla de Dimensión:**VENDEDORES\_EXPANDED

* Personal asignado a ventas.

| **Columna** | **Tipo** | **Descripción** |
| --- | --- | --- |
| id\_vendedor | integer (PK) | ID del vendedor |
| nombre\_vendedor | varchar | Nombre completo |
| id\_sucursal | integer (FK) | Relación a sucursal |
| fecha\_ingreso | date | Desde cuándo trabaja |
| activo | boolean | Estado laboral actual |

**Uso principal:** control de fuerza de ventas, historial, estado, etc.

**5. Tabla de Dimensión:**MEDIOS\_PAGO\_EXPANDED

* Métodos de pago normalizados.

| **Columna** | **Tipo** | **Descripción** |
| --- | --- | --- |
| id\_medio\_pago | integer (PK) | ID del medio |
| nombre\_medio\_pago | varchar | Tipo de pago (Tarjeta, QR, etc.) |

**Uso principal:** estandarización para análisis de preferencias de pago.

**6. Tabla de Hechos:**VENTAS\_EXPANDED

* Registra el encabezado de cada transacción, vinculando un cliente a una fecha y un medio de pago.

| **Columna** | **Tipo** | **Descripción** |
| --- | --- | --- |
| id\_venta | integer (PK) | Identificador único para cada venta |
| fecha | date | Fecha de compra |
| id\_cliente | integer (FK) | Clave foránea para cliente |
| id\_sucursal | integer (FK) | Clave foránea para sucursal |
| id\_vendedor | integer (FK) | Clave foránea para vendedor |
| id\_medio\_pago | integer (FK) | Clave foránea para medio de pago |
| nombre\_cliente | varchar | Snapshot nombre |
| email | varchar | Snapshot email |
| medio\_pago\_original | varchar | Nombre para medio de pago |
| estado\_venta | varchar | Ej: Finalizada, Cancelada |
| monto\_bruto | decimal | Antes de descuentos |
| monto\_neto | decimal | Luego de descuentos |

**Uso principal:** Registrar cada transacción comercial, permitiendo análisis de volumen, ingresos, frecuencia, canal, vendedor y comportamiento del cliente

**4. Tabla de Hechos:**DETALLE\_DE\_VENTAS\_EXPANDED

* Contiene las líneas de cada venta, especificando qué productos, en qué cantidad y a qué precio se vendieron, descuentos aplicados.

| **Columna** | **Tipo** | **Descripción** |
| --- | --- | --- |
| id\_venta | integer (FK) | Clave foránea para venta asociada |
| id\_producto | integer (FK) | Clave foránea de producto comprado |
| nombre\_producto | varchar | Nombre de producto |
| cantidad | integer | Unidades vendidas |
| precio\_unitario | decimal | Precio aplicado |
| importe | decimal | cantidad × precio |
| descuento\_aplicado\_pct | float | Porcentaje de rebaja (0–1) |
| subtotal | decimal | Importe con descuento aplicado |

**Uso principal:** permite **ventas con múltiples productos** y análisis granular.

* **Características del Muestreo**

**Periodo:**  
Las transacciones abarcan del 2 de enero del 2023 al 28 de junio de 2024.  
Los clientes tienen fechas de alta desde enero de 2023, lo que permite análisis de antigüedad, frecuencia de compra fidelización.

**Contexto geográfico:**  
Las ventas ocurren en sucursales de la Tienda Aurelion ubicadas en las ciudades de la provincia de Córdoba, Argentina:  
Carlos Paz, Río Cuarto, Córdoba, Villa María, Alta Gracia y Mendiolaza.  
Cada ciudad cuenta con al menos una sucursal simulada (6), lo que permite análisis por localización.

**Complejidad:**El dataset presenta una complejidad de nivel intermedio-avanzado, incluyendo:

* + Múltiples medios de pago (ahora normalizados).
  + Asignación explícita de vendedores a sucursales.
  + Relación clara entre cliente y ubicación geográfica.
  + Jerarquía de productos con categorías (\*)
  + Tipos de medios de pago.

**Extensibilidad y Aplicación**

Este dataset y su modelo pueden ampliarse con facilidad (fechas, promociones, campañas, ingresos, calendario, inventario) y mantiene total compatibilidad con los datos inciales. Lo que permite una base sólida para ejercicios de análisis exploratorio, creación de dashboards y modelado predictivo orientado a resolver los retos de negocio de la Tienda Aurelion.

### 1.3 Información, pasos, pseudocódigo y diagrama del programa

* **Descripción del funcionamiento:**

El programa del proyecto "Aurelion" es una herramienta de análisis de datos desarrollada en Python, ejecutada a través de Jupyter Notebook, cuyo objetivo es consolidar, limpiar y analizar los datos transaccionales de la tienda. Utilizando siete tablas expandidas (clientes, productos, ventas, detalle\_ventas, sucursales, vendedores, medios\_pago), el programa construye un DataFrame maestro con vista 360°, desde donde se llevan a cabo análisis descriptivos y visualizaciones.

Todo el proceso ocurre con base en una estructura modular cuidadosamente diseñada, el programa ejecuta una serie de análisis predefinidos para responder a preguntas clave del negocio como:

* ¿Cuáles son los productos más vendidos?
* ¿Qué desempeño tiene cada sucursal o vendedor?
* ¿Cuáles son los métodos de pagos más utilizados?
* ¿Qué relación hay entre descuentos y categorías?
* ¿Qué clientes son los que más compran?

Finalmente, presenta estos resultados de forma clara en mediante visualizaciones, sentando las bases para una toma de decisiones informada y basada en datos. Esta etapa representa un análisis centralizado, transparente y reproducible, que sienta las bases para el análisis en modelos predictivos.

* **Pasos del desarrollo:**

El desarrollo del programa siguió una metodología estructurada en cinco pasos clave:

1. **Cargar y Consolidar Datos:**  
   Se desarrolló la lógica inicial para leer los siete archivos de datos utilizando la librería Pandas. En esta fase, se realizaron las tareas de limpieza (manejo de nulos, eliminación de duplicados) y la unión (merge) de todas las tablas para crear un data frame maestro unificado **df\_master**, que contiene toda la información necesaria para el análisis.
2. **Crear Funciones de Búsqueda y Análisis:**  
   El núcleo del análisis se encapsuló en funciones modulares y reutilizables. Se creó una función específica para cada análisis del negocio (ventas por categoría, top productos, ranking de clientes, entre otros) Este enfoque permite que el código sea más limpio, fácil de mantener, reutilizar y validar.
3. **Desarrollo Secuencial en Jupyter Notebook:**

El flujo fue estructurado por secciones secuenciales utilizando Markdown y bloques de código, diseñados para ser ejecutados paso a paso. privilegiando una experiencia exploratoria secuencial muy clara y reproducible, alineada con entornos académicos y orientada a apoyar procesos de análisis y toma de decisiones empresariales.

1. **Validación de Errores:**  
   Se añadieron verificaciones para prevenir errores comunes (archivos faltantes, duplicados, tipos incompatibles). Esto garantiza robustez durante la ejecución, y mejora la experiencia del usuario.
2. **Integración con Copilot:**  
   A lo largo de todo el desarrollo, se utilizó GitHub Copilot como un asistente de programación. Su rol fue fundamental para acelerar la escritura de código repetitivo (boilerplate), sugerir sintaxis para operaciones complejas en Pandas y generar rápidamente funciones útiles para exploración y visualización.

* **Pseudocódigo:**

El pseudocódigo representa la estructura modular y de alto nivel del programa. El proceso principal invoca a una serie de subprocesos, donde cada uno es responsable de una etapa específica del flujo de análisis (ver Anexos). A continuación se presenta un resumen estructurado:

INICIO\_PROGRAMA

// PASO 1: Importar Librerías

IMPORTAR pandas, matplotlib, seaborn

// PASO 2: Cargar tablas

LEER clientes, productos, ventas, detalle\_ventas, sucursales, vendedores y medios\_pago

// PASO 3: Limpiar y revisar datos

REVISAR nulls, tipos y duplicados

// PASO 4: Unificar DataFrames con MERGE encadenado

CREAR df\_master con todas las columnas necesarias

// PASO 5: Crear nuevas columnas

Extraer fechas, calcular montos reales, clasificar por descuento

// PASO 6: Análisis agregados

Productos más vendidos, ingresos por ciudad, mejor vendedor, etc.

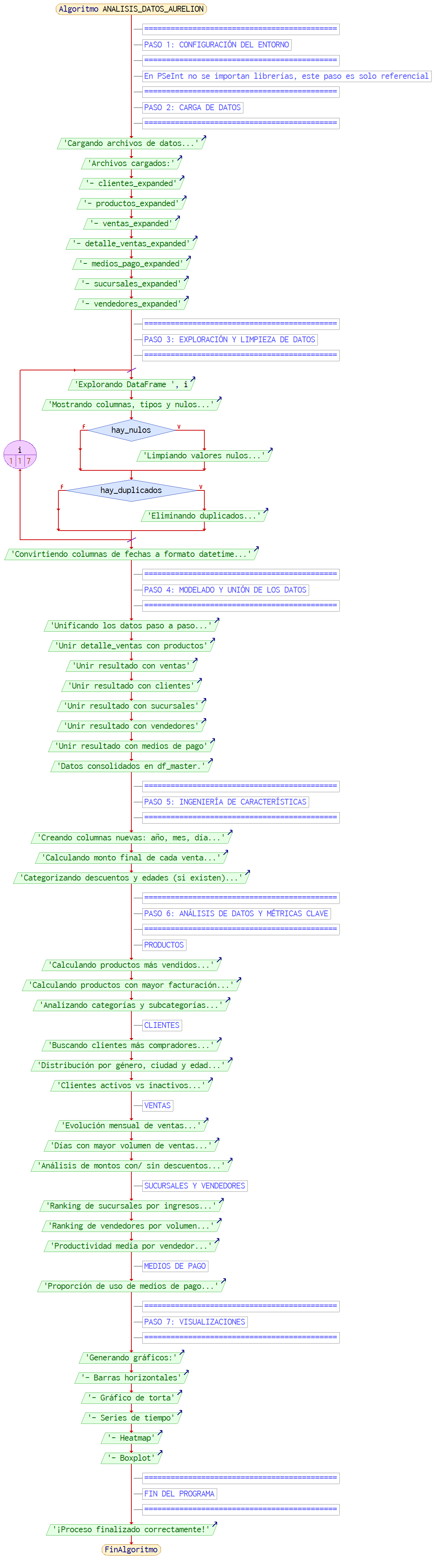
// PASO 7: Visualización

Gráficos de barras, tortas, series de tiempo, boxplots

FIN\_PROGRAMA

* **Diagrama de flujo:**

El siguiente diagrama de flujo, generado en PSeInt, ilustra la secuencia lógica del pseudocódigo modular. Representa el flujo principal del programa, donde cada bloque rectangular con un borde doble simboliza la ejecución de un subproceso que contiene las operaciones detalladas de esa etapa.



### 1.4 Sugerencias y mejoras aplicadas con Copilot

Durante el desarrollo, GitHub Copilot actuó como un asistente proactivo. Sin embargo, su uso requirió un juicio crítico para asegurar que el código final fuera eficiente, legible y se ajustara a los objetivos del proyecto.

* **Sugerencias Aceptadas:**
  + **Autocompletado de Código en Pandas:** Al escribir operaciones de agregación, Copilot demostró ser extremadamente útil. Por ejemplo, después de escribir df\_master.groupby('Categoría'), sugirió correctamente la continuación de la cadena de métodos: ['Importe'].sum().sort\_values(ascending=False). Aceptar esta sugerencia aceleró significativamente el desarrollo del paso de análisis.
  + **Generación de Plantillas para Visualización:** Para la creación de gráficos con Matplotlib y Seaborn, Copilot generó la estructura base (el "boilerplate") de cada gráfico. Tras escribir un comentario como # Gráfico de barras para ventas por ciudad, sugirió el bloque completo sns.barplot(...), plt.title(...), plt.xlabel(...), plt.ylabel(...), lo que permitió centrarse únicamente en la personalización de los detalles estéticos.
* **Sugerencias Descartadas:**
  + **Uso de Funciones Excesivamente Complejas:** En algunas ocasiones, para tareas simples de filtrado, Copilot sugirió el uso de funciones lambda anidadas o list comprehensions complejas. Se decidió descartar estas sugerencias en favor de métodos de Pandas más explícitos y legibles (como el encadenamiento de .loc[] o .groupby()), priorizando la claridad y el fácil mantenimiento del código para un proyecto de este alcance.
  + **Sugerencias Fuera de Alcance:** Al principio del paso de análisis, Copilot sugirió implementar un modelo de machine learning (como K-Means para segmentar clientes) directamente en el script. Aunque es una excelente idea para una fase futura, se descartó porque el objetivo principal de esta etapa inicial es realizar un **análisis descriptivo** por ahora.

## ****Programa en Python****

**Archivo:** proyecto\_aurelion.ipynb

**Tipo:** Jupyter Notebook (\*.ipynb)

* **Funcionalidad:** El notebook ejecuta de forma estructurada el análisis completo de las ventas de Aurelion desde múltiples dimensiones. A partir del df\_master consolidado, permitirá generar:
  + Tablas de resumen (top productos, ingresos por ciudad, vendedores destacados).
  + Visualizaciones automatizadas.
  + Análisis comparativo entre años, meses, ciudades, medios de pago, etc.
* **Interacción**: La interacción con el programa se realiza dentro de un entorno de **Jupyter Notebook (**.ipynb**)**, lo que facilita un análisis exploratorio y documentado, mediante la **ejecución secuencial de celdas** de código y texto, está estructurado como un informe narrativo donde cada paso del proceso (carga, limpieza, análisis, visualización) está encapsulado en su propia celda o conjunto de celdas. La interacción del usuario consiste en:
  + **Leer la Documentación (Markdown):** Cada bloque de código está precedido por una celda de texto (Markdown) que explica el propósito y la lógica de la operación que se va a realizar.
  + **Ejecutar Celdas de Código:** El usuario ejecuta cada celda de código de forma individual y en orden (Shift + Enter).
  + **Observar la Salida Inmediata (Inline):** El resultado de cada operación —ya sea la previsualización de una tabla (.head()), un resumen informativo (.info()) o un gráfico completo— se muestra inmediatamente debajo de la celda que lo generó.
  + **Flexibilidad para modificar:** El entorno permite ajustar filtros, parámetros o funciones sin afectar la estructura general.
* Este enfoque **paso a paso** permite al usuario controlar el ritmo del análisis, inspeccionar el estado de los datos en cada etapa y modificar fácilmente cualquier parámetro para experimentar en tiempo real, lo cual es ideal para el desarrollo y la validación de un flujo de análisis de datos, convirtiendo el notebook tanto en una herramienta de análisis como en una pieza de documentación funcional.

Anexos:

**Pseudocódigo**:

INICIO\_PROGRAMA

// ============================================

// PASO 1: CONFIGURACIÓN DEL ENTORNO

// ============================================

IMPORTAR librerías necesarias:

- pandas

- matplotlib

- seaborn

- numpy //(operaciones estadísticas)

// ============================================

// PASO 2: CARGA DE DATOS

// ============================================

CARGAR los siguientes DataFrames desde archivo o ruta:

- clientes\_expanded

- productos\_expanded

- ventas\_expanded

- detalle\_ventas\_expanded

- medios\_pago\_expanded

- sucursales\_expanded

- vendedores\_expanded

// ============================================

// PASO 3: EXPLORACIÓN Y LIMPIEZA DE DATOS

// ============================================

PARA cada tabla EN la lista de DataFrames:

MOSTRAR columna, tipos de datos y valores nulos

SI existen valores nulos:

APLICAR estrategia de limpieza (rellenar, eliminar filas, etc.)

SI existen duplicados:

ELIMINAR duplicados si corresponde

FIN PARA

CONVERTIR columnas de fechas (fecha, fecha\_alta, fecha\_ingreso) al tipo datetime

// ============================================

// PASO 4: MODELADO Y UNIÓN DE LOS DATOS

// ============================================

// Se realizarán las uniones necesarias para formar un DataFrame maestro

UNIR detalle\_ventas\_expanded CON productos\_expanded mediante id\_producto ⇒ df\_dv\_prod

UNIR df\_dv\_prod CON ventas\_expanded mediante id\_venta ⇒ df\_ventas\_completas

UNIR df\_ventas\_completas CON clientes\_expanded mediante id\_cliente ⇒ df\_vtas\_clientes

UNIR df\_vtas\_clientes CON sucursales\_expanded mediante id\_sucursal ⇒ df\_vtas\_sucursal

UNIR df\_vtas\_sucursal CON vendedores\_expanded mediante id\_vendedor ⇒ df\_vtas\_vendedor

UNIR df\_vtas\_vendedor CON medios\_pago\_expanded mediante id\_medio\_pago ⇒ df\_master

// df\_master contendrá ahora información consolidada de cliente, producto, venta, sucursal, vendedor y medio de pago

// ============================================

// PASO 5: INGENIERÍA DE CARACTERÍSTICAS (FEATURE ENGINEERING)

// ============================================

CREAR nuevas columnas:

- Año de venta (a partir de fecha)

- Mes de venta

- Día de la semana

- Rango\_etario\_categorizado (opcional, basado en edad\_rango)

- Descuento\_aplicado\_pct\_categorizado (segmentar nivel de descuentos)

- Monto\_final = subtotal - (subtotal \* descuento\_aplicado\_pct)

// ============================================

// PASO 6: ANÁLISIS DE DATOS Y MÉTRICAS CLAVE

// ============================================

// ANÁLISIS DE PRODUCTOS

CALCULAR: Productos más vendidos (por cantidad total)

CALCULAR: Productos con mayor facturación (por subtotal o monto\_final)

CALCULAR: Categorías y subcategorías más populares

// ANÁLISIS DE CLIENTES

CALCULAR: Clientes que más compran (por monto\_total)

CALCULAR: Distribución por género, ciudad, edad

CALCULAR: Clientes inactivos vs activos

// ANÁLISIS DE VENTAS

ANALIZAR: Evolución de ventas por mes

ANALIZAR: Días de la semana con mayor volumen de ventas

ANALIZAR: Comparativa monto\_bruto vs monto\_neto, con descuentos

// ANÁLISIS DE SUCURSALES Y VENDEDORES

CALCULAR: Ranking sucursales por ingresos

CALCULAR: Vendedores con mayor volumen de ventas

CALCULAR: Productividad media por vendedor

// ANÁLISIS DE MEDIOS DE PAGO

CALCULAR: Proporción de medios de pago utilizados

// ============================================

// PASO 7: VISUALIZACIONES Y TABLEROS

// ============================================

GENERAR gráficos:

- Barras horizontales (Top productos, Top clientes)

- Torta (distribución medios de pago, productos por categoría)

- Serie de tiempo (ventas por mes)

- Calor o heatmap (ventas por sucursal y mes)

- Boxplot (descuentos distribuidos por categoría o ciudad)

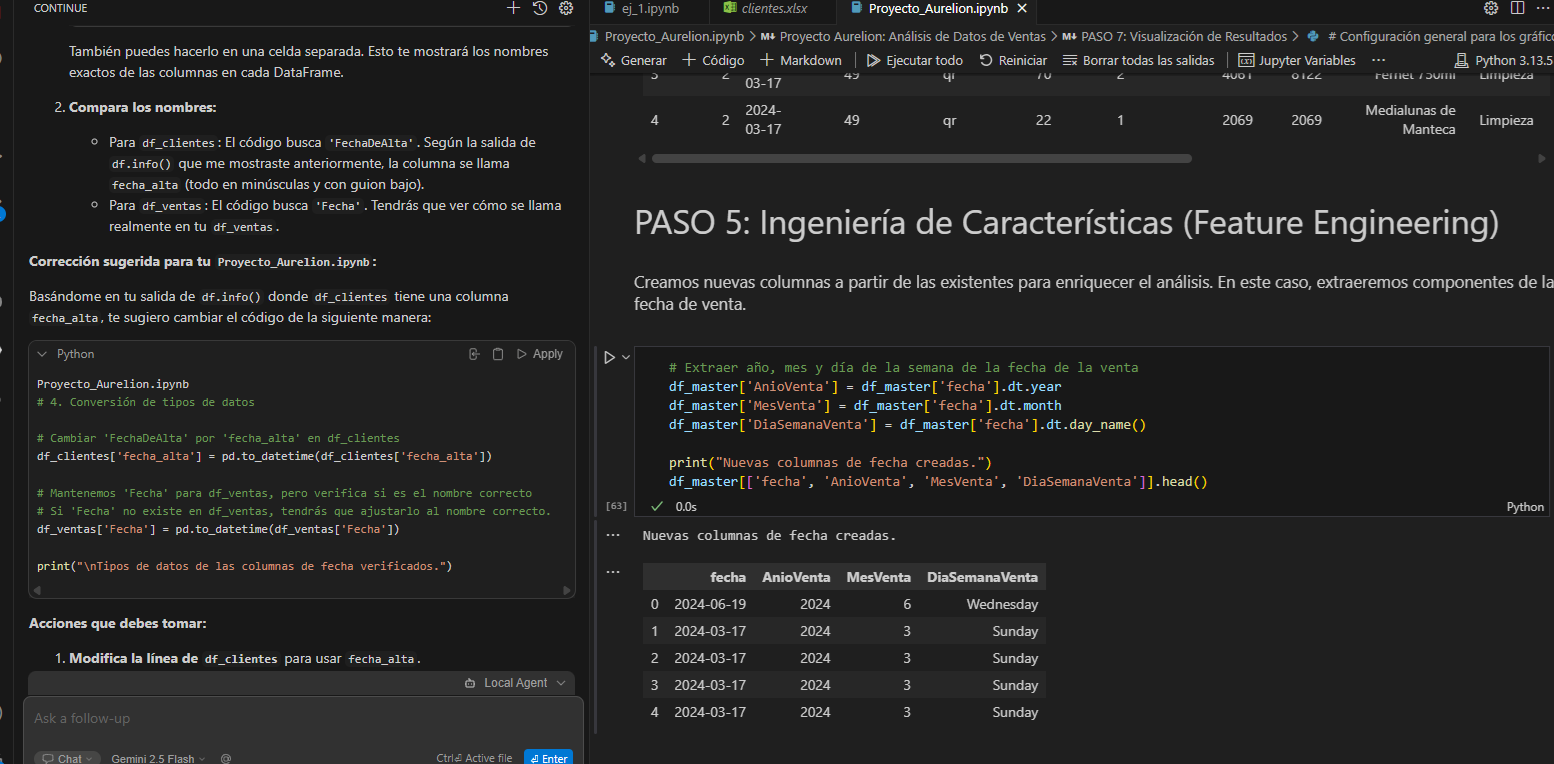
// ============================================

// FIN DEL PROGRAMA

// ============================================

FINALIZAR\_PROGRAMA

**Sugerencias de Copilot**



.

