Assesment Data Driven Consultant

Omni.Pro

Realizado por Ing. Gabriel García Montoya

Linkedin: https://www.linkedin.com/in/gabyx7677/

Este proyecto cuenta con respaldo en
Github:https://github.com/gabyx7677/data_insights_consulting

Sección 1: Manejo de Bases de Datos

`data_insights_consulting`

Objetivo

Esta sección tiene como propósito evaluar las habilidades prácticas en el manejo de bases de datos utilizando SQL. Se busca demostrar la capacidad de realizar consultas, emplear funciones de agregación, manejar múltiples tablas y generar vistas que faciliten el análisis de la información.

Descripción

Se cuenta con una base de datos relacional que contiene información de clientes, pedidos, productos y detalles de los pedidos. A partir de estas tablas se desarrollarán las siguientes tareas:

1. Determinar la cantidad total de productos vendidos por cada categoría.
2. Calcular el total acumulado gastado por cada cliente en el tiempo, ordenado por fecha del pedido.
3. Identificar los productos que nunca han sido vendidos.
4. Crear una vista llamada `customer_order_summary` que muestre:
    - Nombre del cliente
    - Número total de pedidos
    - Total gastado

Los resultados se mostrarán en tablas y se incluirá una interpretación para cada consulta.

Carga de datos

Se utiliza un archivo `.sqlite` para simular un entorno de base de datos real, permitiendo ejecutar consultas SQL directamente y mantener las relaciones entre tablas sin procesos previos de carga o transformación.

Estructura de tablas

CUSTOMERS

- **id**: Identificador único del cliente.
- **name**: Nombre completo del cliente.
- **email**: Correo electrónico del cliente.

- **created_at**: Fecha de registro del cliente.

ORDERS

- **id**: Identificador único del pedido.
- **customer_id**: ID del cliente que realizó el pedido.
- **order_date**: Fecha en que se realizó el pedido.
- **total_amount**: Monto total del pedido.

ORDER ITEMS

- **id**: Identificador único del detalle del pedido.
- **order_id**: ID del pedido al que pertenece el artículo.
- **product_id**: ID del producto vendido.
- **quantity**: Cantidad de unidades vendidas.
- **unit_price**: Precio unitario del producto.

PRODUCTS

- **id**: Identificador único del producto.
- **name**: Nombre del producto.
- **category**: Categoría del producto.

In [1]:
```
import sqlite3
import pandas as pd

# Conexión a la base
conn = sqlite3.connect('../datasets/sqlite/base_datos.sqlite')

# Lectura de las tablas
customers = pd.read_sql_query("SELECT * FROM customers", conn)
orders = pd.read_sql_query("SELECT * FROM orders", conn)
order_items = pd.read_sql_query("SELECT * FROM order_items", conn)
products = pd.read_sql_query("SELECT * FROM products", conn)
```

In [2]:
```
customers
```

Out[2]:

	id	name	email	created_at
0	1	Juan Pérez	juan@email.com	2023-01-15
1	2	Laura Gómez	laura@email.com	2023-02-10

In [3]:
```
orders
```

Out[3]:

	id	customer_id	order_date	total_amount
0	1	1	2023-04-10	150.0
1	2	2	2023-05-20	200.0
2	3	1	2023-06-05	75.0

In [4]:
```
order_items
```

Out[4]:

	id	order_id	product_id	quantity	unit_price
0	1	1	10	2	50.0
1	2	1	11	1	50.0
2	3	2	12	4	50.0

In [5]:
```
products
```

Out[5]:

	id	name	category
0	10	Laptop	Electrónica
1	11	Mouse	Electrónica
2	12	Silla Oficina	Muebles

1. Productos vendidos por categoría

Se determina la cantidad total de productos vendidos por cada categoría utilizando las tablas `order_items` y `products`. El objetivo es identificar cuáles categorías han tenido mayor rotación en ventas.

In [6]:
```
# Se calcula la cantidad total de productos vendidos por categoría
query_cat_sales = """
SELECT
    p.category AS categoria,
    SUM(oi.quantity) AS total_vendidos
FROM order_items oi
JOIN products p ON oi.product_id = p.id
GROUP BY categoria
ORDER BY total_vendidos DESC;
"""

cat_sales = pd.read_sql_query(query_cat_sales, conn)
cat_sales
```

Out[6]:

	categoria	total_vendidos
0	Muebles	4
1	Electrónica	3

Conclusión:
Se observa que la categoría con mayor número de unidades vendidas es `Muebles`, seguida de `Electrónica`.
Este resultado sugiere que, dentro del periodo analizado, la demanda se concentra en artículos de mobiliario, lo que podría estar vinculado a campañas comerciales, estacionalidad o cambios en las necesidades del cliente.
Contar con este tipo de indicadores permite priorizar la gestión de inventarios, optimizar la asignación de recursos y enfocar las estrategias de marketing en las categorías de mayor rotación, sin descuidar aquellas con menor participación para identificar oportunidades de crecimiento.

2. Total acumulado gastado por cliente

Se agrega una columna que muestra el gasto acumulado de cada cliente en el tiempo, ordenado por fecha del pedido. Esto permite identificar patrones de consumo y clientes de alto valor.

In [7]:
```
# Se calcula el gasto acumulado por cliente ordenado por fecha
query_cumulative = """
SELECT
    c.name AS cliente,
    o.order_date AS fecha_pedido,
    o.total_amount AS monto_pedido,
    SUM(o.total_amount) OVER (PARTITION BY c.id ORDER BY o.order_date) AS gasto_acu
FROM orders o
JOIN customers c ON o.customer_id = c.id
ORDER BY cliente, fecha_pedido;
"""

cumulative = pd.read_sql_query(query_cumulative, conn)
cumulative
```

Out[7]:

	cliente	fecha_pedido	monto_pedido	gasto_acumulado
0	Juan Pérez	2023-04-10	150.0	150.0
1	Juan Pérez	2023-06-05	75.0	225.0
2	Laura Gómez	2023-05-20	200.0	200.0

Conclusión:
Se logra visualizar cómo evoluciona el gasto de cada cliente a lo largo del tiempo,

identificando no solo el monto total invertido, sino también la frecuencia y el comportamiento de compra.

Este indicador es esencial para el diseño de programas de fidelización, ya que permite segmentar clientes según su valor acumulado, anticipar necesidades y priorizar acciones comerciales hacia aquellos con mayor potencial de recompra.
Además, facilita la detección temprana de clientes inactivos, lo que abre la puerta a estrategias de reactivación específicas.

3. Productos nunca vendidos

Se identifican los productos que no tienen registros en la tabla `order_items`, lo que permite detectar oportunidades para optimizar el catálogo.

In [8]:
```sql
# Productos que nunca han sido vendidos
query_unsold = """
SELECT
    p.id,
    p.name,
    p.category
FROM products p
LEFT JOIN order_items oi ON p.id = oi.product_id
WHERE oi.product_id IS NULL;
"""

unsold = pd.read_sql_query(query_unsold, conn)
unsold
```

Out[8]:

id	name	category

Conclusión:
La consulta confirma que, en el periodo y dataset analizado, **no existen productos sin ventas registradas**.
Esto indica que, al menos en esta muestra, la rotación de inventario es completa y no se evidencian referencias inactivas.

En un contexto real, este resultado serviría como punto de control positivo, aunque se recomienda ejecutar esta validación periódicamente para anticipar posibles acumulaciones de inventario o problemas de comercialización.

4. Vista: `customer_order_summary`

Se crea una vista que consolida información clave por cliente: nombre, número total de pedidos y monto total gastado.

In [9]:
```sql
# Creación de la vista
create_view = """
```

```
CREATE VIEW customer_order_summary AS
SELECT
    c.name AS cliente,
    COUNT(o.id) AS total_pedidos,
    SUM(o.total_amount) AS total_gastado
FROM customers c
JOIN orders o ON c.id = o.customer_id
GROUP BY c.name;
"""
conn.execute("DROP VIEW IF EXISTS customer_order_summary")
conn.execute(create_view)

# Consultar la vista
summary = pd.read_sql_query("SELECT * FROM customer_order_summary", conn)
summary
```

Out[9]:

	cliente	total_pedidos	total_gastado
0	Juan Pérez	2	225.0
1	Laura Gómez	1	200.0

Conclusión:

La vista `customer_order_summary` consolida información clave de cada cliente, mostrando el número total de pedidos y el monto total gastado.
En este caso, se observa que **Juan Pérez** ha realizado 2 pedidos por un valor acumulado de 225, mientras que **Laura Gómez** ha realizado 1 pedido por un valor de 200.

Este tipo de vista es útil para alimentar reportes recurrentes y dashboards comerciales, así como para segmentar clientes según su volumen de compras, priorizar esfuerzos de fidelización y detectar perfiles con mayor potencial de retención o ventas adicionales.

In [ ]:

SECCIÓN 2: Analítica web y evaluación técnica de DataLayer

Objetivo:
Evaluar la implementación del dataLayer y el flujo de eventos en el sitio super99.com mediante inspección en DevTools.

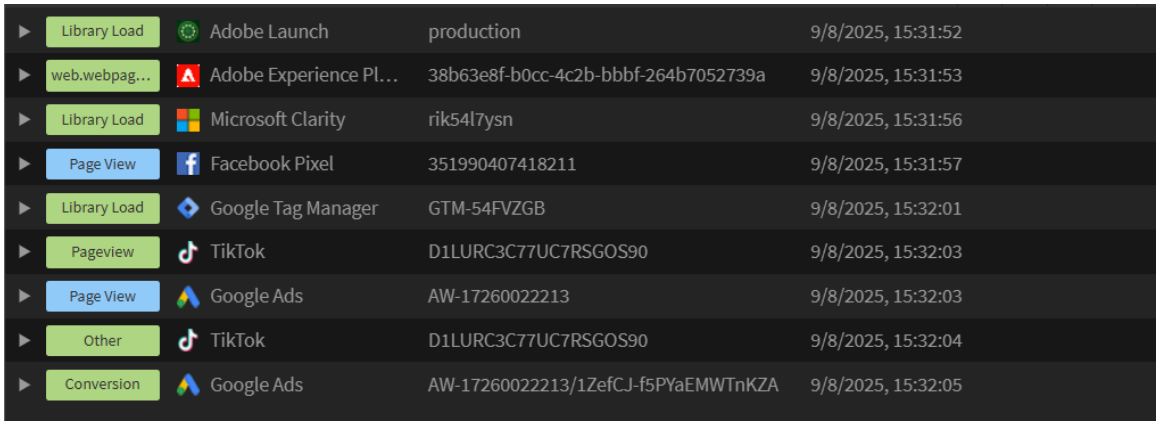
A. Verificación de envío de eventos

Se inspeccionó el objeto **window.dataLayer** en distintas interacciones del usuario:

1. Carga de página y envío de eventos iniciales

Al cargar la página https://www.super99.com, se observan en el `dataLayer` eventos estándar de Google Tag Manager (`gtm.js`, `gtm.dom`, `gtm.load`) que indican la correcta inicialización y ejecución del contenedor GTM.

Aunque no se detecta un evento explícito `pageview` dentro del `dataLayer`, la inspección con la extensión Omnibug muestra que las plataformas de analítica externas (Facebook Pixel, TikTok Pixel, Google Ads) reciben correctamente el evento de visita a la página (`pageview`), evidenciando un disparo exitoso.



Adicionalmente, se observan cargas de librerías y eventos de Adobe Experience Platform, lo que confirma una integración múltiple y adecuada.

Etiquetas detectadas en el sitio:

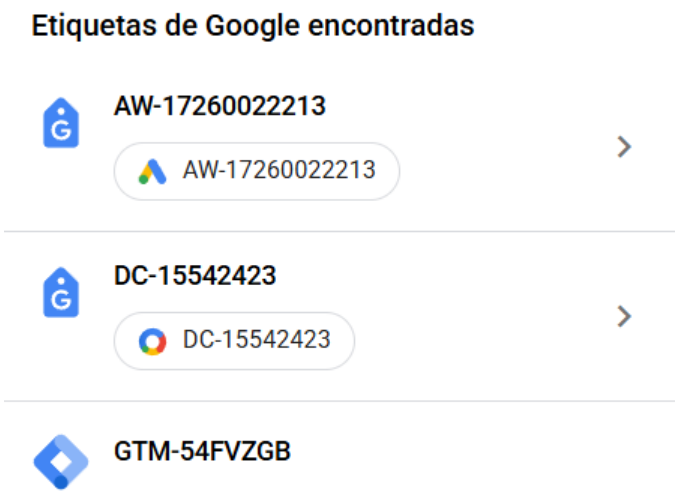
Mediante la extensión **Tag Assistant** se identificaron las siguientes etiquetas activas en la página:

- Google Ads (AW-17260022213): Para seguimiento de conversiones y remarketing.

- DoubleClick (DC-15542423): Plataforma de marketing para gestión y remarketing.

- Google Tag Manager (GTM-54FVZGB): Contenedor que centraliza la gestión de etiquetas y eventos.

Estas etiquetas están correctamente implementadas y activas, evidenciando una configuración integral para la medición y optimización de campañas publicitarias.



2. Visualización de página de detalle de producto

Se ingresó a un producto cualquiera, en este caso MAGIA BLANCA BLANQ. REG 4/1GLN.

Al acceder a la página de detalle de producto, el `dataLayer` registra un evento específico denominado `productDetail`, que indica que la vista del producto ha sido cargada correctamente. Este evento es complementario a los eventos estándar de Google Tag Manager (`gtm.js`, `gtm.dom`, `gtm.load`) que gestionan la carga y el disparo de etiquetas.

```
> const eventos = window.dataLayer.map(e => e.event).filter(Boolean);
  console.log('Eventos únicos en dataLayer:', [...new Set(eventos)]);

  Eventos únicos en dataLayer:  (4) ['gtm.js', 'productDetail', 'gtm.dom', 'gtm.load']   VM3084:2
                                    0: "gtm.js"
                                    1: "productDetail"
                                    2: "gtm.dom"
                                    3: "gtm.load"
                                    length: 4
                                  ▸ [[Prototype]]: Array(0)

< undefined

> temp1

< 'productDetail'

> copy(JSON.stringify(temp1, null, 2))

< undefined
```

Se verifica el evento productDetail

```
> const detalleProducto = window.dataLayer.find(e => e.event === 'productDetail');
  console.log(detalleProducto);

  ▾ {event: 'productDetail', ecommerce: {…}, gtm.uniqueEventId: 4}   VM3100:2
    ▸ ecommerce: {detail: {…}, impressions: Array(0)}
      event: "productDetail"
      gtm.uniqueEventId: 4
    ▸ [[Prototype]]: Object

< undefined

> temp2

< ▸ {event: 'productDetail', ecommerce: {…}, gtm.uniqueEventId: 4}

> copy(JSON.stringify(temp2, null, 2))

< undefined
```

Json resultante:


```
{
  "event": "productDetail",
  "ecommerce": {
    "detail": {
      "products": [
        {
          "id": "20114381",
          "name": "MAGIA BLANCA BLANQ. REG 4/1GLN",
          "category": ""
        }
      ]
    },
    "impressions": []
  },
  "gtm.uniqueEventId": 4
}
```

Análisis técnico del evento `productDetail`

El evento `productDetail` registrado en el `dataLayer` contiene un objeto `ecommerce` con detalles específicos del producto visualizado. En particular, incluye un arreglo `products` que detalla:

- **ID:** `20114381`

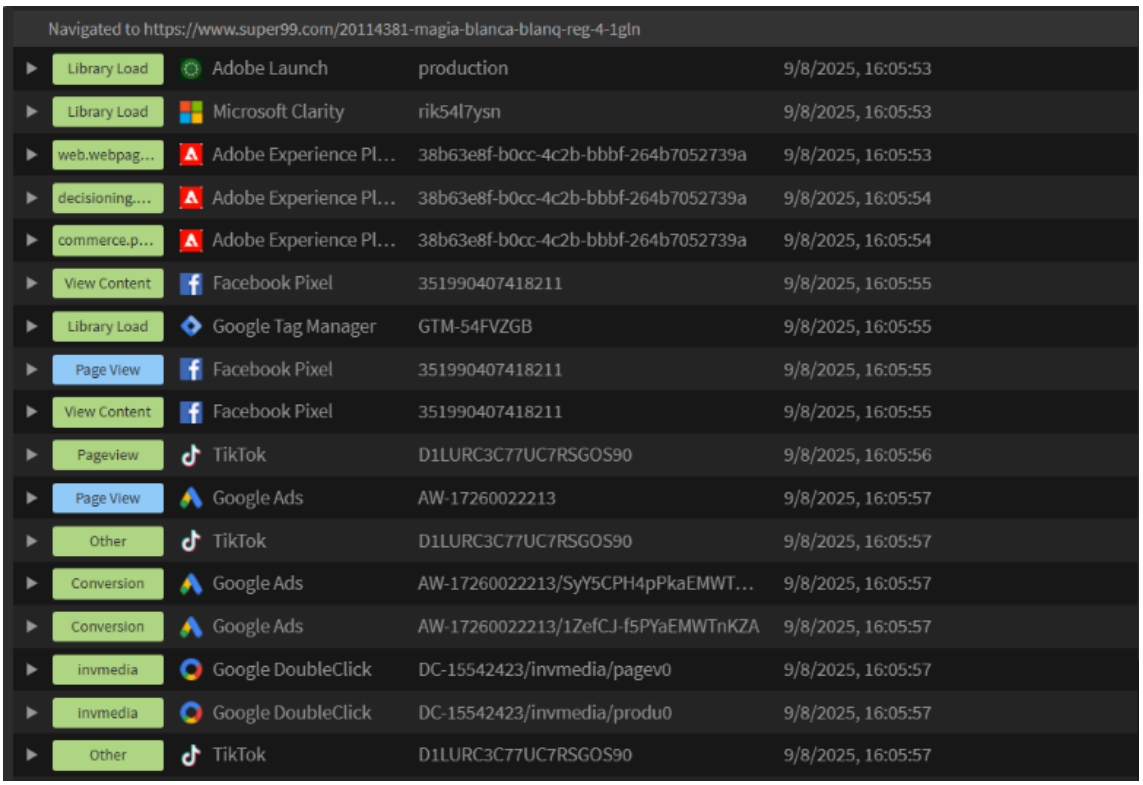
- **Nombre:** `MAGIA BLANCA BLANQ. REG 4/1GLN`

- **Categoría:** (vacía, se recomienda completar para optimizar segmentación).

Esta estructura estándar permite a las plataformas de analítica capturar con precisión la interacción del usuario con productos específicos. El campo `gtm.uniqueEventId` garantiza la unicidad del evento dentro de Google Tag Manager.

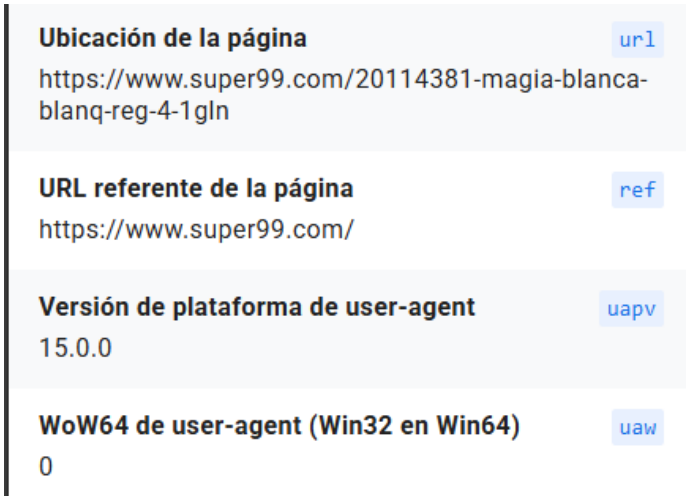
Esta implementación es sólida, aunque se sugiere completar la categoría para mejorar la calidad del análisis y segmentación futura.

Se visualiza que la plataforma Adobe Experience Platform recibe un evento `commerce.productViews` que incluye datos del producto como el detergente "Magia Blanca", validando que la información de visualización está siendo enviada correctamente.



En Omnibug se constataron eventos de `pageview` para Facebook Pixel y Adobe, así como eventos de Google DoubleClick (`invmedia`) y carga de librería para Microsoft Clarity, confirmando la integración multicanal y la correcta transmisión de eventos.

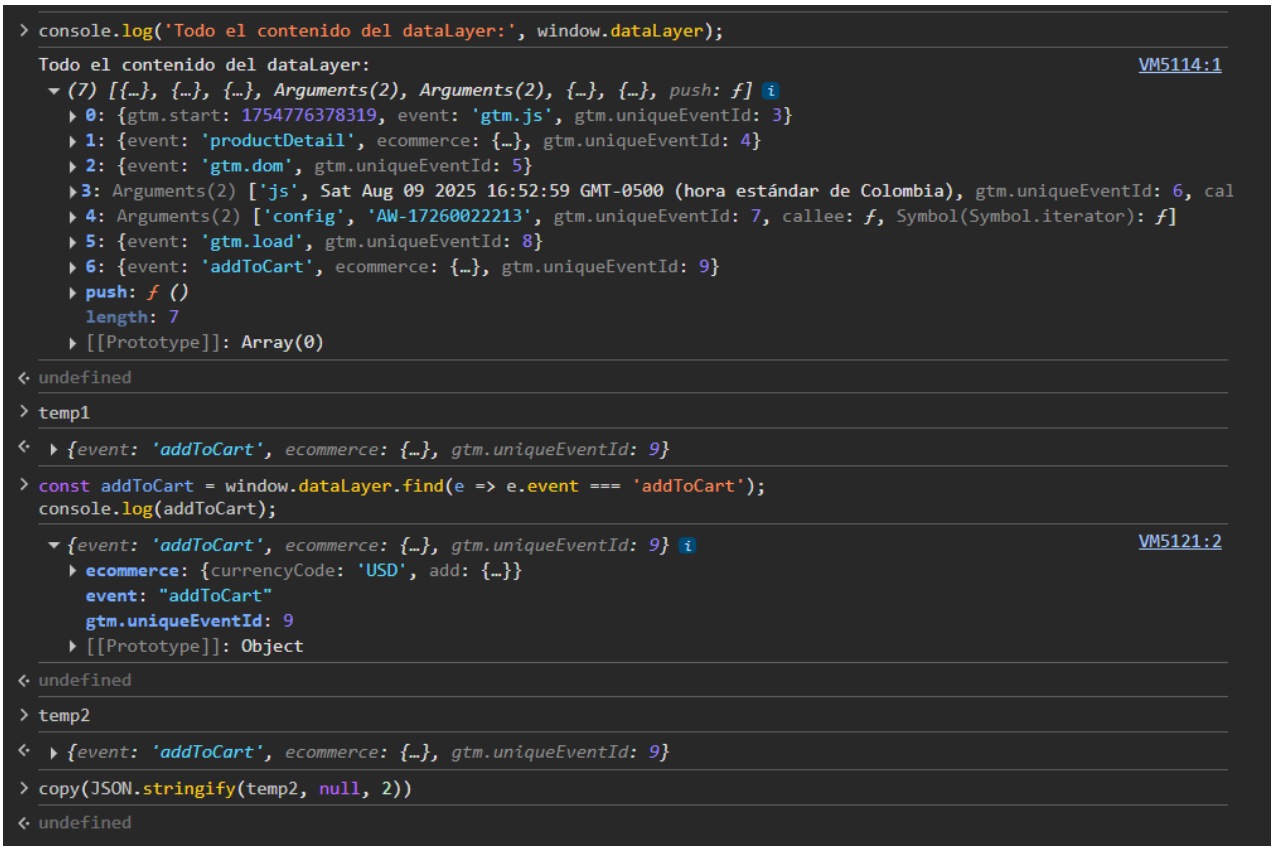
Finalmente, Tag Assistant continúa mostrando activas las etiquetas de Google Ads (AW-17260022213), DoubleClick (DC-15542423) y Google Tag Manager (GTM-54FVZGB), garantizando la continuidad en la medición y remarketing.



3. Añadir al carrito

Análisis técnico del evento `addToCart`

Al añadir un producto al carrito, el `dataLayer` registra el evento `addToCart`, que incluye un objeto `ecommerce` con detalles específicos de la acción de añadir.



```
{
  "event": "addToCart",
  "ecommerce": {
    "currencyCode": "USD",
    "add": {
      "products": [
        {
          "id": "20114381",
          "name": "MAGIA BLANCA BLANQ. REG 4/1GLN",
          "price": 3.35,
          "quantity": 3
        }
      ]
    }
  },
  "gtm.uniqueEventId": 9
}
```


- Nombre: MAGIA BLANCA BLANQ. REG 4/1GLN

- Precio unitario: 3.35

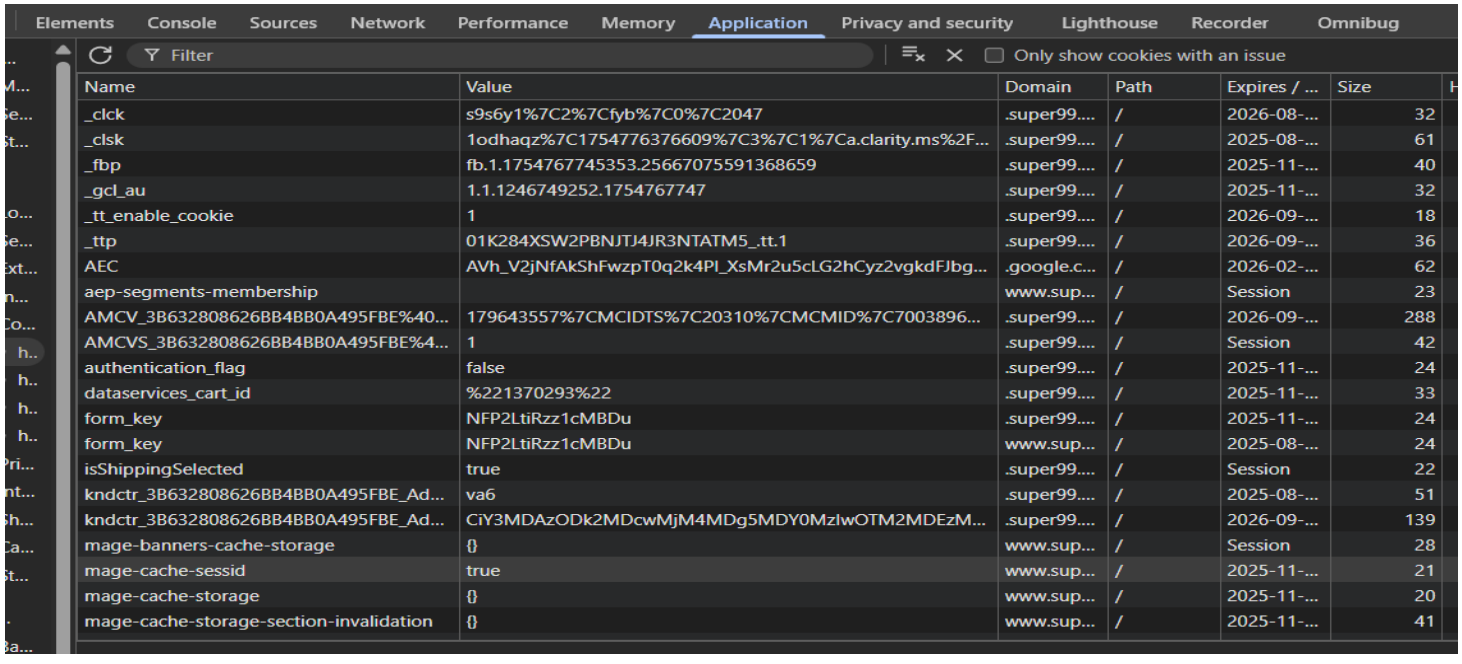
- Cantidad: 2

El campo `gtm.uniqueEventId` identifica de manera única este evento en el flujo de Google Tag Manager.

La información capturada permite medir con precisión qué productos y en qué cantidades son añadidos al carrito, lo cual es crucial para análisis de comportamiento de compra y para activar campañas de remarketing.

La validación en Omnibug y Tag Assistant confirma que este evento se está enviando correctamente a las plataformas de analítica, asegurando la integridad y continuidad del seguimiento en el funnel de conversión.

B. Identificadores en la cookies



Name	Value	Domain	Path	Expires / ...	Size
_clck	s9s6y1%7C2%7Cfyb%7C0%7C2047	.super99....	/	2026-08-...	32
_clsk	1odhaqz%7C1754776376609%7C3%7C1%7Ca.clarity.ms%2F...	.super99....	/	2025-08-...	61
_fbp	fb.1.1754767745353.25667075591368659	.super99....	/	2025-11-...	40
_gcl_au	1.1.1246749252.1754767747	.super99....	/	2025-11-...	32
_tt_enable_cookie	1	.super99....	/	2026-09-...	18
_ttp	01K284XSW2PBNJTJ4JR3NTATM5_.tt.1	.super99....	/	2026-09-...	36
AEC	AVh_V2jNfAkShFwzpT0q2k4PI_XsMr2u5cLG2hCyz2vgkdFJbg...	.google.c...	/	2026-02-...	62
aep-segments-membership		www.sup...	/	Session	23
AMCV_3B632808626BB4BB0A495FBE%40...	179643557%7CMCIDTS%7C20310%7CMCMID%7C7003896...	.super99....	/	2026-09-...	288
AMCVS_3B632808626BB4BB0A495FBE%4...	1	.super99....	/	Session	42
authentication_flag	false	.super99....	/	2025-11-...	24
dataservices_cart_id	%221370293%22	.super99....	/	2025-11-...	33
form_key	NFP2LtiRzz1cMBDu	.super99....	/	2025-11-...	24
form_key	NFP2LtiRzz1cMBDu	www.sup...	/	2025-08-...	24
isShippingSelected	true	.super99....	/	Session	22
kndctr_3B632808626BB4BB0A495FBE_Ad...	va6	.super99....	/	2025-08-...	51
kndctr_3B632808626BB4BB0A495FBE_Ad...	CiY3MDAzODk2MDcwMjM4MDg5MDY0MzIwOTM2MDEzM...	.super99....	/	2026-09-...	139
mage-banners-cache-storage	{}	www.sup...	/	Session	28
mage-cache-sessid	true	www.sup...	/	2025-11-...	21
mage-cache-storage	{}	www.sup...	/	2025-11-...	20
mage-cache-storage-section-invalidation	{}	www.sup...	/	2025-11-...	41

Durante la inspección del almacenamiento web se identificaron varios identificadores persistentes, entre los más relevantes:

- _fbp: Cookie de Facebook Pixel que permite rastrear y segmentar usuarios para campañas de remarketing en Facebook Ads.

- _clsk y _clk: Cookies asociadas a Microsoft Clarity, que facilitan el análisis del comportamiento del usuario y seguimiento de sesiones.

- dataservices_cart_id: Identificador relacionado con la gestión del carrito de compras, útil para mantener el estado del usuario durante la sesión.

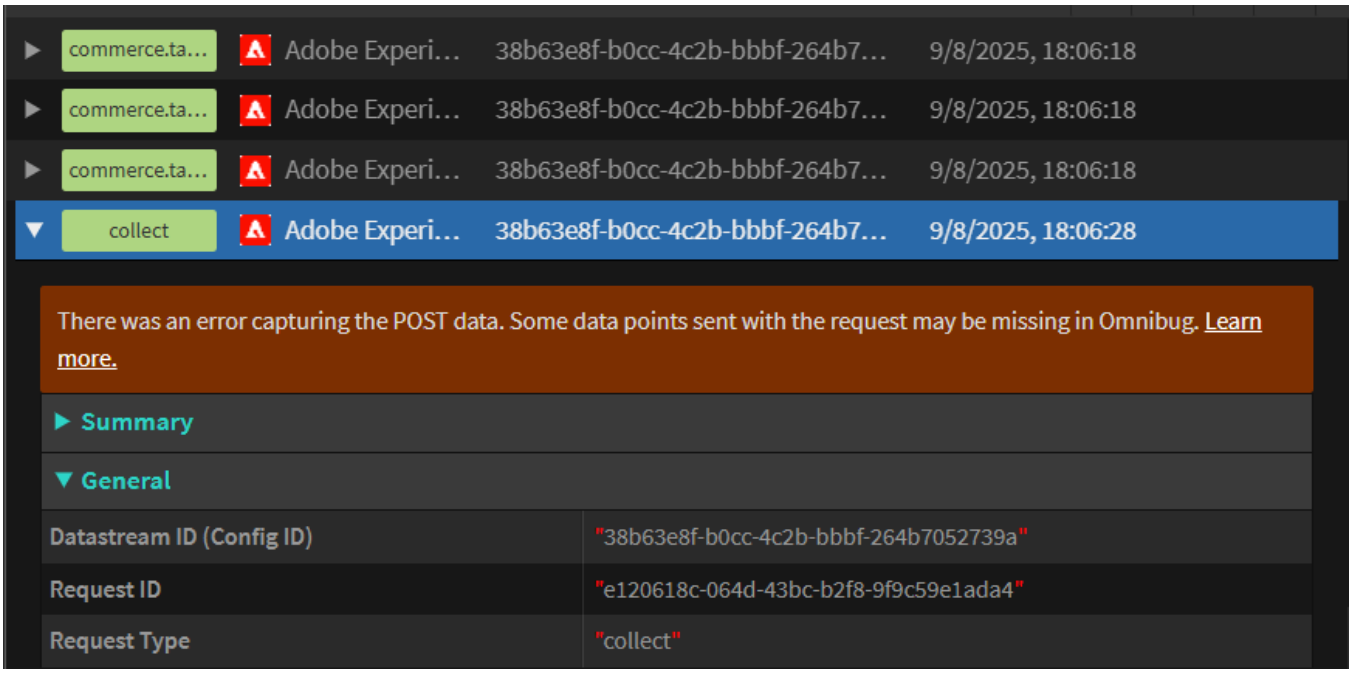
- form_key, authentication_flag, isShippingSelected`: Variables que indican estado y seguridad en formularios y procesos de compra.

No se observaron cookies clásicas de Google Analytics como `_ga`, `_gid` o `_gat`, lo que puede indicar una configuración diferente o el uso de plataformas alternativas de análisis como Adobe Experience Platform y Microsoft Clarity.

Esta variedad de identificadores permite un seguimiento integral y multicanal del comportamiento del usuario, optimizando la capacidad de personalización y remarketing.

C. Errores o inconsistencias detectadas

- En el payload del evento productDetail enviado a Adobe Experience Platform, el campo category se encuentra vacío (""), lo que implica pérdida de información clave para segmentación y análisis por tipo de producto.



- **Categorías de producto incompletas**: En productDetail el campo category aparece vacío, reduciendo la capacidad de segmentación.
- **Ausencia de evento pageview en el dataLayer**: Aunque se envía a plataformas externas, no hay un registro estandarizado interno que facilite integraciones y depuración.
- **Falta de cookies estándar de GA**: No se detectan _ga o _gid, lo que sugiere dependencia de Adobe y Clarity y dificulta compatibilidad con GA.
- **Posible duplicidad de seguimiento**: Integraciones múltiples (Facebook, TikTok, DoubleClick, Adobe) sin gobernanza clara pueden provocar eventos duplicados y afectar rendimiento.
- **Datos incompletos en addToCart**: No incluye categoría, marca o inventario.
- **Exceso de peticiones de tracking**: En *Network* se observan múltiples llamadas repetitivas a collect y tp2 desde clarity.js y Magento Event Collector, generando sobrecarga y posible redundancia por falta de batching u optimización en el envío de datos.

D. Mejoras propuestas para el dataLayer

- Ejemplo de evento productDetail de otro producto en el dataLayer con campos vacíos o faltantes (category, brand, etc.). Completar estos valores mejoraría la calidad de los datos para segmentación y marketing, se visualiza que es un error generalizado en varios productos y una oportunidad de mejora.


```
{
  "event": "productDetail",
  "ecommerce": {
    "detail": {
      "products": [
        {
          "id": "20136999",
          "name": "LA CROIX SPRKING CHERRY BLOSSOM 24/12OZ",
          "category": ""
        }
      ]
    },
    "impressions": []
  },
  "gtm.uniqueEventId": 4
}
```

1. **Completar campos faltantes**: Rellenar siempre category, brand, variant y otros atributos relevantes tanto en productDetail como en addToCart para optimizar segmentación.
2. **Estandarizar eventos de navegación**: Añadir un evento pageview consistente en el dataLayer para todas las páginas, de forma que cualquier herramienta de analítica pueda consumirlo sin depender de integraciones específicas.
3. **Incluir identificadores unificados**: Añadir un userId anónimo persistente que permita vincular interacciones de un mismo usuario entre sesiones y plataformas.
4. **Optimizar el firing de etiquetas**: Revisar condiciones de disparo para evitar redundancias entre Facebook, TikTok, DoubleClick y Adobe que puedan degradar el rendimiento del sitio.
5. **Agregar eventos adicionales del funnel**: Como beginCheckout, purchase, removeFromCart y viewCategory para tener una trazabilidad completa del comportamiento de compra.
6. **Documentar la estructura del dataLayer**: Mantener un documento de referencia para el equipo de marketing y desarrollo que estandarice nombres, jerarquía y atributos, evitando inconsistencias en futuras implementaciones.

En general, la implementación actual del dataLayer en super99.com permite un seguimiento funcional y multicanal de la actividad del usuario, pero presenta oportunidades claras de optimización en la completitud de datos, estandarización de eventos y eficiencia en el envío de tracking. Las mejoras propuestas contribuirían a una medición más precisa, segmentaciones más efectivas y un menor impacto en el rendimiento del sitio.

SECCIÓN 3: MODELO DE ENTIDAD-RELACIÓN

Objetivo: Evaluar la capacidad para elaborar un modelo de entidad-relación (ER) dadas distintas fuentes de datos.

Resumen: La compañía cuenta con múltiples fuentes de datos valiosos (formularios de HubSpot, datos de navegación web, CRM y POS) que no han sido integrados para monetizar su valor. El objetivo es identificar posibles identificadores clave para unificarlos mediante una CDP (Customer Data Platform) y proponer un esquema de entidad-relación que soporte la unificación de perfiles.

1. Fuentes de datos y posibles identificadores:

A. Formularios de HubSpot (Leads) - Email (identificador primario) - Nombre y Apellido - Teléfono - ID interno de HubSpot - Fecha de registro - Fuente de captación (campaña/medio)

B. Navegación, comportamiento e intención de compra (Sitios web) - Cookie ID o Session ID (persistente si es posible) - Dirección IP - Device ID (si disponible) - Email (si el usuario inició sesión) - Historial de páginas vistas, clics y productos visitados

C. Datos de clientes y ventas desde el CRM - ID Cliente (interno CRM) - Email - Teléfono - Nombre y Apellido - Dirección física - Historial de compras

D. Información transaccional desde el POS - Ticket/Factura ID - ID Cliente POS (si existe) - Email o Teléfono (si se capturó en la compra) - Fecha y hora de la transacción - SKU/Producto comprado - Monto total

2. Estrategia de unificación en CDP: El CDP funcionará como repositorio central que consolida perfiles únicos a partir de reglas de coincidencia (matching rules) sobre los identificadores. Ejemplo de jerarquía de prioridad de emparejamiento:

A. Email como identificador universal (siempre que sea válido)

B. Teléfono como identificador secundario

C. ID internos de CRM/POS como referencias de sistemas fuente

D. Device ID o Cookie ID para tráfico anónimo

3. Modelo de Entidad-Relación

1. Entidad Cliente

- **Clave primaria (PK):** ClienteID
- **Atributos:** DocumentoID, Email, FechaRegistro, FuentePreferida, Nombre, Teléfono.

- **Relaciones:**
    - **1 a muchos** con Venta (ClienteID → ClienteID FK).
    - **1 a 1** con Profile (ClienteID → CRMClienteID en Profile).

2. Entidad Lead

- **PK:** LeadID
- **Atributos:** Email, FechaCreacion, Nombre, Origen, Teléfono.
- **Relaciones:**
    - **1 a 1** con Profile (LeadID → HubSpotLeadID en Profile).

3. Entidad Profile

- **PK:** ProfileID
- **Atributos:** ConsentFlags, CRMClienteID (FK), Emails (array), HubSpotLeadID (FK), LastSeen, LTV_estimate, MergeAudit, Otros_Datos, Phones (array), SourceIDs.
- **Relaciones:**
    - **1 a 1** con Cliente (CRMClienteID FK).
    - **1 a 1** con Lead (HubSpotLeadID FK).
    - **1 a muchos** con Navegacion_Event (ProfileID → ProfileID FK).

4. Entidad Navegacion_Event

- **PK:** NavegacionID
- **Atributos:** CookieID / ClienteID, Evento, Page, ProductID, ProfileID (FK), Timestamp, URL, UsuarioIdentificado (ClienteID o LeadID, nullable).
- **Relaciones:**
    - **Muchos a 1** con Profile (ProfileID FK).

5. Entidad Venta

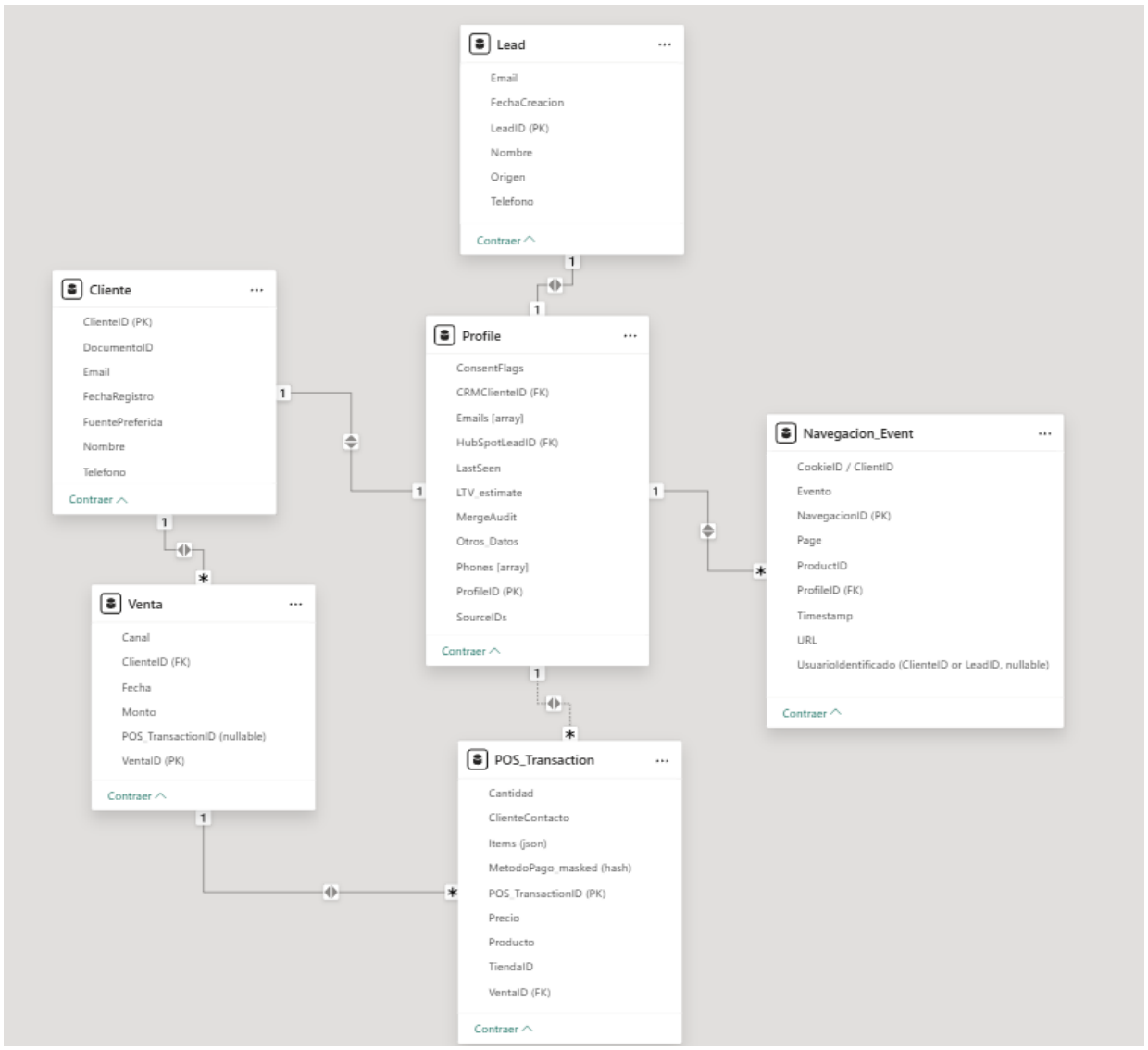
- **PK:** VentaID

- **Atributos:** Canal, ClienteID (FK), Fecha, Monto, POS_TransactionID (nullable).
- **Relaciones:**
    - **Muchos a 1** con Cliente (ClienteID FK).
    - **1 a muchos** con POS_Transaction (VentaID → VentaID FK).

6. Entidad POS_Transaction

- **PK:** POS_TransactionID
- **Atributos:** Cantidad, ClienteContacto, Items (json), MetodoPago_masked, Precio, Producto, TiendaID, VentaID (FK).
- **Relaciones:**
    - **Muchos a 1** con Venta (VentaID FK).

Diagrama ER:



4. Beneficios esperados:

- **Visión 360° del cliente:** Integración de datos de ventas, navegación, leads y POS en un solo perfil unificado.

- **Capacidad de personalización de campañas:** Mensajes y ofertas adaptadas a intereses y comportamientos detectados.

- **Mayor precisión en atribución de ventas:** Posibilidad de rastrear el viaje completo del cliente desde el primer contacto hasta la compra.

- **Incremento en la conversión por segmentación avanzada:** Creación de segmentos dinámicos basados en hábitos reales y datos históricos.

- **Detección temprana de oportunidades de venta cruzada (cross-selling) y upselling:** Identificación de productos o servicios complementarios con mayor probabilidad de compra.

- **Optimización de inversión en marketing:** Reducción de gasto en campañas poco efectivas gracias a mejor focalización.

- **Mejora en retención de clientes:** Identificación de patrones de abandono y aplicación de estrategias preventivas.

- **Seguimiento omnicanal:** Análisis del comportamiento del cliente en todos los canales (online, físico, CRM, web).

- **Integración con analítica avanzada e IA:** Posibilidad de aplicar modelos predictivos para pronosticar comportamientos de compra.

- **Base sólida para automatización de marketing:** Activación de campañas automáticas basadas en eventos o triggers en tiempo real.

SECCIÓN 4: CASO DE CONSULTORÍA

Acciones propuestas para resolver los problemas de retraso en reportes y falta de centralización de información:

1. **Diseñar e implementar un Data Warehouse corporativo**
    - Unificar todas las fuentes de datos (ERP, CRM, inventarios, canales digitales, POS) en un repositorio único optimizado para análisis.
    - Esto elimina silos, asegura coherencia en los indicadores y facilita el acceso controlado por perfiles de usuario.
2. **Automatizar el flujo ETL con procesos escalables**
    - Definir pipelines de extracción, transformación y carga que operen en intervalos cortos o en tiempo real.
    - Usar entornos distribuidos y paralelismo (por ejemplo, procesamiento en clúster y jobs programados) para manejar grandes volúmenes sin retrasos.
    - Incorporar validaciones automáticas de calidad de datos para prevenir errores antes de llegar al reporte.
3. **Desplegar dashboards interactivos y de actualización continua**
    - Construir paneles de control conectados directamente al Data Warehouse, con filtros dinámicos y métricas clave actualizadas sin intervención manual.
    - Permitir a cada área acceder solo a la información relevante para agilizar la toma de decisiones y reducir dependencia de reportes solicitados por correo.
4. **Definir un modelo de gobernanza de datos**
    - Establecer políticas claras de calidad, seguridad y uso de la información.
    - Definir responsables (“data owners” y “data stewards”) para garantizar que la información sea confiable y consistente a lo largo de toda la organización.
5. **Implementar monitoreo y alertas de datos**
    - Configurar sistemas que detecten anomalías en tiempo real (por ejemplo, caídas abruptas en ventas o retrasos en cargas de datos).
    - Integrar notificaciones automáticas para que los equipos actúen antes de que los problemas afecten la operación.