Diplomatura en Ciencia de Datos

TP Módulo 1: Data Warehousing & OLAP.

Profesor: Dr. Alejandro Vaisman

Grupo N°2

Integrantes:

- Valeria Natalie Bassano
- Ivan Kim
- Belén Barceló
- Martina del Castillo
- Federico Pupillo

1. Contexto y Descripción del Problema

Byhjeans es una empresa especializada en la venta de indumentaria femenina ubicada en el barrio de Flores, CABA, Argentina La empresa se fundó en el año 1997, comenzando como un negocio familiar que hoy en día cuenta con seis empleados. La compañía trabaja con producción local en talleres textiles y con importación desde China, lo cual define en gran medida su estructura de costos, estrategias de pricing y posicionamiento en el mercado.

Byhjeans opera con un modelo de negocios mixto en donde el 30% de las ventas vienen de una modalidad B2B desde el único local a la calle con el que cuenta la empresa, mientras que el 70% se concentra en un segmento B2C vía canal digital en Mercado Libre. Los medios de pago difieren entre los dos canales; el local de venta minorista acepta pagos en efectivo y en transferencia únicamente mientras que la venta mayorista tiene un medio de pago a través de Mercado Pago, en donde el cliente solo tiene la opción de dinero en cuenta.

En el primer semestre del 2025, la empresa “Byhjeans”, notó una disminución considerable en el volumen de ventas en comparación al año 2024. Esta caída se explica por el contexto económico del país, caracterizado por una inflación elevada que afectó tanto a los costos de adquisición de mercadería como al comportamiento de consumo de los clientes. Consecuentemente, el aumento en los costos produjo una disminución en la rentabilidad de la empresa.

2. Objetivos

Objetivo principal: Incrementar el volumen de ventas de Byhjeans, empresa de indumentaria femenina

Frente al escenario del primer semestre de 2025, en donde se notó una disminución considerable de ventas, nuestro objetivo es incrementar la recaudación aumentando el volumen de ventas en ambos canales.

Se necesita medir rápido qué está pasando por mes, categoría, color y promociones, y accionar tácticas como bundles, empuje de prendas extras, y descuentos en fines de semana o liquidaciones.

Subobjetivos y Consultas

1. **Aumentar el valor del ticket promedio (AOV).** Llevar el AOV neto un +15% comparando al S1 2024, manteniendo margen bruto. Se mide con ventas netas ÷ tickets

    - ¿Cuál es el ticket promedio general?
    - ¿Cuál es el ticket promedio por producto?
    - ¿Cuál es el ticket promedio por temporada?
    - ¿Cuál es el ticket promedio por dia de semana?
    - ¿Cuál es el ticket promedio por canal de venta?
    - ¿Cuál es el porcentaje de ventas por canal?

2. **Subir unidades por ticket (UPT)**. Aumentar cantidad impulsando combos “remeras”, “‘jeans”, “tapados”, “blusas”, “polleras“, “shorts” y “conjunto lencería”. Se mide con unidades totales ÷ tickets

    - ¿Cuántas unidades se venden por ticket?
    - ¿Cuántas unidades se venden de cada producto?
    - ¿Cuántas unidades se venden de cada producto por temporada?
    - ¿Cuántas unidades se venden de cada producto por día?
    - ¿Cuántas unidades se venden de cada producto por canal de venta?

3. **Elevar “attach rate” de prenda superior**. Lograr que al menos 20% de los tickets incluyan ≥1 prenda superior. Se mide como tickets con accesorios ÷ tickets

    - ¿Qué porcentaje de los tickets totales lleva al menos una prenda superior?
    - ¿Cuál es el AOV de los tickets con al menos una prenda superior?

4. **Efectividad de promociones**. Lograr un AOV con promoción mayor o igual al AOV sin promo. Aumentar cantidad de tickets en fines de semana.

    - ¿Cuántas unidades se vendieron de los distintos modelos de producto con promoción y sin promoción?
    - ¿Qué porcentaje de las ventas se hacen con promoción?
    - ¿Cuál es el ticket promedio con promoción?
    - ¿Cuál es la cantidad de tickets promedio por día los fines de semana?
    - ¿Cuál es la cantidad de tickets promedio por día los días de semana?

5. Aumentar en un +15% la **recurrencia de clientes digitales identificados**, midiendo la proporción de clientes con más de una compra en el semestre

    - ¿Qué porcentaje de los clientes hace más de una compra por semestre en Mercado Libre?
    - ¿Qué porcentaje de clientes tiene 1 compra, 2 compras, 3 compras, 4+ compras en Mercado Libre?

- ¿Cuál es el AOV de un cliente recurrente?

3. Diseño Conceptual

Diseño conceptual tipo estrella:

Guarda las métricas transaccionales (hechos medibles):

- **PK fact_id**
- **FK id_tiempo → Dim_Tiempo**
- **FK id_producto → Dim_Producto**
- **FK id_cliente → Dim_Cliente**
- **FK id_local → Dim_Local**
- **FK id_canal → Dim_CanalVenta**
- **FK id_medio_pago → Dim_MedioPago**
- **FK id_promocion → Dim_Promocion**
- **ticket_id (degenerada)**

Métricas (medidas):

- ➢ **unidades**
- ➢ **importe_bruto**
- ➢ **descuento**
- ➢ **importe_neto**
- ➢ **precio_unitario_neto**

Dimensión 1: Dim_Tiempo

Describe la fecha y jerarquías temporales.

- **PK id_tiempo**
- **fecha (UNIQUE)**
- **dia**
- **mes**
- **trimestre**
- **semestre**
- **anio**
- **es_fin_de_semana (booleano)**

Dimensión 2: Dim_Producto

Describe el artículo vendido.

- **PK id_producto**
- **sku (código interno o externo)**
- **modelo**
-
- **categoria**
- **talle**
- **color**
- **marca**

- **temporada**

Dimensión 3: Dim_Cliente

Describe al cliente

- **PK id_cliente**
- **cuil_cuit**
- **nombre**
- **apellido**
- **fecha_nacimiento**
- **genero**
- **vigencia_inicio (útil para cambios históricos)**

Dimensión 4: Dim_CanalVenta

Identifica el canal de venta.

- **PK id_canal**
- **tipo_canal (ej: “FÍSICO”, “ECOMMERCE”)**

Dimensión 5: Dim_MedioPago

Describe cómo se paga.

- **PK id_medio_pago**
- **tipo_mediopago**

Dimensión 6: Dim_Local

Ubicación física del local de ventas.

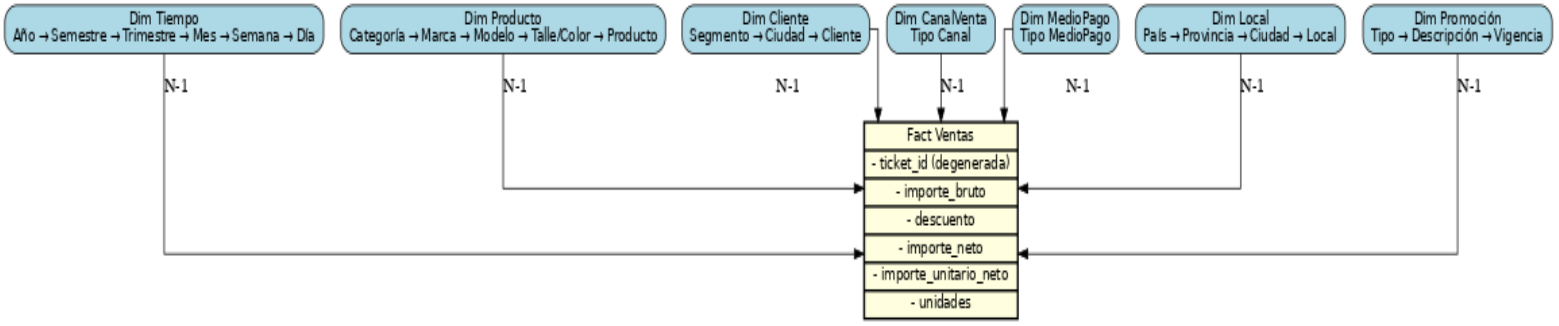
- **PK id_local**
- **nombre_local**
- **ciudad**
- **provincia**
- **pais**

Dimensión 7: Dim_Promocion

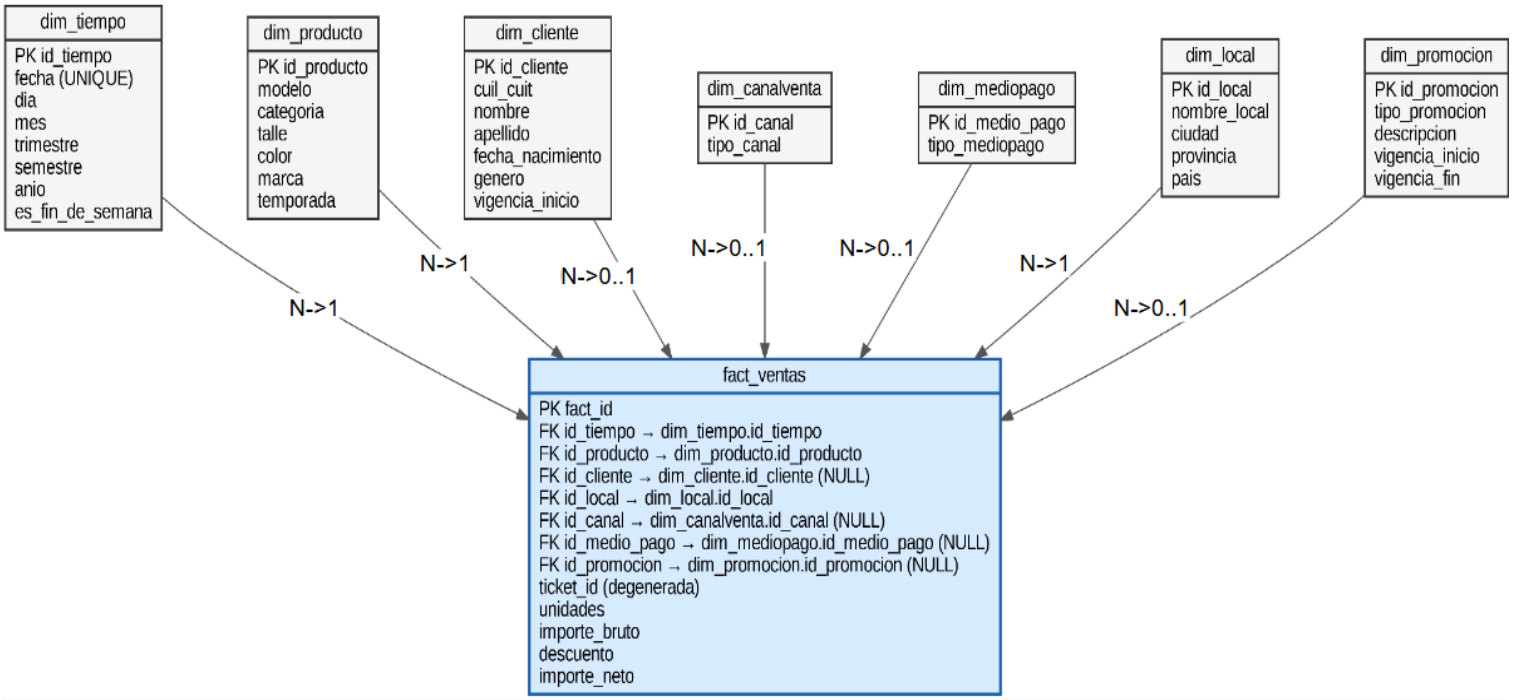
Describe promociones aplicadas.

- **PK id_promocion**
- **tipo_promocion (ej: “Descuento %”, “2x1”)**
- **descripcion**
- **vigencia_inicio**
- **vigencia_fin**

Modelo Conceptual:



4. Diseño Lógico: Estrella



Importe_unitario_neto .

5. Creación del data mart en PostgreSQL - Script

-- DATA MART DE VENTAS - MODELO ESTRELLA (PostgreSQL)

```
-- 1) Esquema

DROP SCHEMA IF EXISTS dw CASCADE;
CREATE SCHEMA dw;
COMMENT ON SCHEMA dw IS 'Esquema para el Data Mart de Ventas';

=====================================================================
-- 2) DIMENSIONES

-- 2.1 Dimensión Tiempo
CREATE TABLE dim_tiempo (
   id_tiempo        BIGINT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
   fecha            DATE NOT NULL UNIQUE,
   dia              SMALLINT NOT NULL CHECK (dia BETWEEN 1 AND 31),
   mes              SMALLINT NOT NULL CHECK (mes BETWEEN 1 AND 12),
   trimestre        SMALLINT NOT NULL CHECK (trimestre BETWEEN 1 AND 4),
   semestre         SMALLINT NOT NULL CHECK (semestre IN (1,2)),
   anio             SMALLINT NOT NULL CHECK (anio BETWEEN 1900 AND 2100),
   es_fin_de_semana  BOOLEAN NOT NULL DEFAULT FALSE
);

-- 2.2 Dimensión Producto
CREATE TABLE dim_producto (
   id_producto   BIGINT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
   sku           TEXT NOT NULL UNIQUE,
   modelo        TEXT,
   categoria     TEXT,
   talle         TEXT,
   color         TEXT,
   marca         TEXT,
   temporada     TEXT
);

-- 2.3 Dimensión Cliente
CREATE TABLE dim_cliente (
   id_cliente       BIGINT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
   cuil_cuit        TEXT,
   nombre           TEXT,
   apellido         TEXT,
   fecha_nacimiento  DATE,
   genero           TEXT,
   segmento         TEXT,
```

```
    ciudad            TEXT,
    vigencia_inicio   DATE
);

-- 2.4 Dimensión Canal de Venta
CREATE TABLE dim_canalventa (
    id_canal   BIGINT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
    tipo_canal TEXT NOT NULL CHECK (tipo_canal IN ('fisico','ecommerce'))
);

-- 2.5 Dimensión Medio de Pago
CREATE TABLE dim_mediopago (
    id_medio_pago   BIGINT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
    tipo_mediopago  TEXT NOT NULL   -- ejemplo: 'efectivo','tarjeta','transferencia'
);

-- 2.6 Dimensión Local
CREATE TABLE dim_local (
    id_local      BIGINT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
    nombre_local  TEXT,
    ciudad        TEXT,
    provincia     TEXT,
    pais          TEXT
);

-- 2.7 Dimensión Promoción
CREATE TABLE dim_promocion (
    id_promocion     BIGINT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
    tipo_promocion   TEXT,     -- bundle / liquidación / cupón / etc.
    descripcion      TEXT,
    vigencia_inicio  DATE,
    vigencia_fin     DATE
);

--
================================================================
-- 3) TABLA DE HECHOS


CREATE TABLE fact_ventas (
    fact_id           BIGINT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,

    id_tiempo         BIGINT NOT NULL REFERENCES dim_tiempo(id_tiempo) ON UPDATE
RESTRICT ON DELETE RESTRICT,
    id_producto       BIGINT NOT NULL REFERENCES dim_producto(id_producto) ON
UPDATE RESTRICT ON DELETE RESTRICT,
    id_local          BIGINT NOT NULL REFERENCES dim_local(id_local)     ON UPDATE
RESTRICT ON DELETE RESTRICT,
```

```
  id_cliente        BIGINT     REFERENCES dim_cliente(id_cliente)       ON UPDATE
RESTRICT ON DELETE RESTRICT,
  id_canal          BIGINT     REFERENCES dim_canalventa(id_canal)      ON UPDATE
RESTRICT ON DELETE RESTRICT,
  id_medio_pago     BIGINT     REFERENCES dim_mediopago(id_medio_pago)  ON
UPDATE RESTRICT ON DELETE RESTRICT,
  id_promocion      BIGINT     REFERENCES dim_promocion(id_promocion)   ON UPDATE
RESTRICT ON DELETE RESTRICT,

  ticket_id         TEXT,
  unidades              INTEGER      NOT NULL CHECK (unidades >= 0),
  importe_bruto         NUMERIC(12,2) NOT NULL CHECK (importe_bruto >= 0),
  descuento             NUMERIC(12,2) NOT NULL DEFAULT 0 CHECK (descuento >= 0),
  importe_neto          NUMERIC(12,2) NOT NULL CHECK (importe_neto >= 0),
  precio_unitario_neto  NUMERIC(12,4) NOT NULL CHECK (precio_unitario_neto >= 0)
);

-- Índices adicionales para performance en joins/filters
CREATE INDEX ON fact_ventas (id_tiempo);
CREATE INDEX ON fact_ventas (id_producto);
CREATE INDEX ON fact_ventas (id_local);
CREATE INDEX ON fact_ventas (id_cliente);
CREATE INDEX ON fact_ventas (id_canal);
CREATE INDEX ON fact_ventas (id_medio_pago);
CREATE INDEX ON fact_ventas (id_promocion);
CREATE INDEX ON fact_ventas (ticket_id);
```

==========

6. Población de las tablas y consultas SQL

1) Poblar Dimensión Tiempo (un año completo)

```
-- Tiempo: 2024/2025
INSERT INTO dim_tiempo (fecha, dia, mes, trimestre, semestre, anio, es_fin_de_semana)
SELECT  d::date                                  AS fecha,
       EXTRACT(DAY   FROM d)::smallint           AS dia,
       EXTRACT(MONTH FROM d)::smallint           AS mes,
       EXTRACT(QUARTER FROM d)::smallint         AS trimestre,
       CASE WHEN EXTRACT(QUARTER FROM d)::int <= 2 THEN 1 ELSE 2 END AS
semestre,
       EXTRACT(YEAR  FROM d)::smallint           AS anio,
       CASE WHEN EXTRACT(ISODOW FROM d)::int IN (6,7) THEN TRUE ELSE FALSE
END AS es_fin_de_semana
FROM generate_series(date '2024-09-17', date '2025-09-17', interval '1 day') AS g(d);
```

2.1 Canal de venta

```
INSERT INTO dim_canalventa (tipo_canal)
VALUES ('fisico'), ('ecommerce');
```

2.2 Medio de pago

```
INSERT INTO dim_mediopago (tipo_mediopago)
VALUES ('efectivo'), ('tarjeta_debito'), ('transferencia'), ('mercadopago');
```

2.3 Locales

```
INSERT INTO dim_local (nombre_local, ciudad, provincia, pais) VALUES
('Recoleta',       'Buenos Aires', 'Buenos Aires', 'Argentina'),
('Córdoba Centro', 'Córdoba',      'Córdoba',      'Argentina'),
('Rosario Shop',   'Rosario',      'Santa Fe',     'Argentina'),
('Mendoza Mall',   'Mendoza',      'Mendoza',      'Argentina');
```

2.4 Promociones (fechas de vigencia)

```
INSERT INTO dim_promocion (tipo_promocion, descripcion, vigencia_inicio, vigencia_fin)
VALUES
('bundle',       'Vestido + Accesorio -15%',  '2025-07-01', '2025-07-31'),
('fin_de_semana','-20% sábados y domingos',   '2025-08-01', '2025-08-31'),
('liquidacion',  'Temporada Invierno -30%',   '2025-09-01', '2025-09-30'),
('cupon',        'Cupón OTOÑ-10%',      '2025-05-01', '2025-05-31');
```

3) Poblar Dimensión Producto

```
WITH cat AS (SELECT unnest(ARRAY['remeria','jean','vestido','accesorios']) AS categoria),
    marca AS (SELECT unnest(ARRAY['Urban','Neo','Vintage','Sport']) AS marca),
    modelo AS (SELECT unnest(ARRAY['A1','B2','C3','D4','E5','F6']) AS modelo),
    talle  AS (SELECT unnest(ARRAY['XS','S','M','L','XL']) AS talle),
    color  AS (SELECT unnest(ARRAY['negro','blanco','azul','rojo','verde']) AS color),
    tempo  AS (SELECT unnest(ARRAY['SS25','FW24']) AS temporada),
    base AS (
      SELECT row_number() OVER () AS rn,
            c.categoria, m.marca, mo.modelo, t.talle, co.color, te.temporada
      FROM cat c CROSS JOIN marca m CROSS JOIN modelo mo
      CROSS JOIN talle t CROSS JOIN color co CROSS JOIN tempo te
    )
INSERT INTO dim_producto (sku, modelo, categoria, talle, color, marca, temporada)
SELECT
  'SKU' || lpad(rn::text, 5, '0') AS sku,
  modelo, categoria, talle, color, marca, temporada
FROM base
ORDER BY rn
LIMIT 200;
```

4) Poblar Dimensión Cliente (sintética)

```
WITH nombres AS (SELECT
unnest(ARRAY['Ana','Luis','María','Sofía','Pedro','Elena','Julián','Valentina','Diego','Carla']) AS
nombre),
    apes    AS (SELECT
unnest(ARRAY['García','Rodríguez','López','Martínez','Pérez','Sánchez','Romero','Gómez'])
AS apellido),
    gen     AS (SELECT unnest(ARRAY['F','M','X']) AS genero),
    seg     AS (SELECT unnest(ARRAY['alta','media','baja']) AS segmento),
    ciu     AS (SELECT unnest(ARRAY['Buenos Aires','Córdoba','Rosario','Mendoza','La
Plata']) AS ciudad),
    base AS (
      SELECT row_number() OVER () AS rn,
           n.nombre, a.apellido, g.genero, s.segmento, c.ciudad
      FROM nombres n, apes a, gen g, seg s, ciu c
    )
INSERT INTO dim_cliente (cuil_cuit, nombre, apellido, fecha_nacimiento, genero,
segmento, ciudad, vigencia_inicio)
SELECT
  '20' || lpad(rn::text, 8, '0') || '3' AS cuil_cuit,
  nombre, apellido,
  date '1980-01-01' + (random()*15000)::int, -- nacimientos entre 1980 y ~2021
  genero,
  segmento,
  ciudad,
  date '2025-01-01'
FROM base
ORDER BY rn
LIMIT 1000;
```

5) Poblar Fact_Ventas

Período: S2 2025 (julio–diciembre)

```
WITH
dias AS (
  SELECT id_tiempo, fecha
  FROM dim_tiempo
  WHERE fecha BETWEEN DATE '2025-01-01' AND DATE '2025-07-31'
),
tickets_por_dia_local AS (
  SELECT d.id_tiempo, d.fecha, l.id_local,
         (20 + (random()*40)::int) AS tickets_en_dia      -- entre 20 y 60 tickets por local y día
  FROM dias d CROSS JOIN dim_local l
),
tickets AS (
  SELECT
    t.id_tiempo, t.fecha, t.id_local,
```

```
    generate_series(1, t.tickets_en_dia) AS nro_ticket
  FROM tickets_por_dia_local t
),
lineas AS (
  SELECT
    tk.id_tiempo, tk.fecha, tk.id_local,
    concat('T', to_char(tk.fecha,'YYYYMMDD'), '-', lpad(tk.nro_ticket::text, 5, '0')) AS ticket_id,
    generate_series(1, 1 + (random()*2)::int) AS nro_linea,          -- 1 a 3 líneas
    (1 + (random()*2)::int) AS unidades,                             -- 1 a 3 unidades
    (SELECT id_producto    FROM dim_producto    ORDER BY random() LIMIT 1) AS
id_producto,
    CASE WHEN random() < 0.90 THEN
       (SELECT id_cliente FROM dim_cliente ORDER BY random() LIMIT 1)
    END AS id_cliente,
    (SELECT id_canal FROM dim_canalventa ORDER BY random() LIMIT 1) AS id_canal,
    (SELECT id_medio_pago  FROM dim_mediopago  ORDER BY random() LIMIT 1) AS
id_medio_pago,
    (SELECT id_promocion
       FROM dim_promocion pr
      WHERE (pr.vigencia_inicio IS NULL OR pr.vigencia_inicio <= tk.fecha)
        AND (pr.vigencia_fin    IS NULL OR pr.vigencia_fin    >= tk.fecha)
      ORDER BY random()
      LIMIT 1
    ) AS id_promocion_vigente,
   (15 + random()*85)::numeric AS precio_lista
  FROM tickets tk
),
medidas AS (
  SELECT
    l.id_tiempo, l.id_producto, l.id_local, l.id_cliente, l.id_canal, l.id_medio_pago,
    CASE WHEN random() < 0.15 THEN l.id_promocion_vigente ELSE NULL END AS
id_promocion,
    l.ticket_id,
    l.unidades,
    l.precio_lista,
    (l.precio_lista * l.unidades)::numeric(12,2) AS importe_bruto,
    CASE
      WHEN l.id_promocion_vigente IS NULL OR random() >= 0.15 THEN 0::numeric(12,2)
      ELSE (l.precio_lista * l.unidades) * (0.05 + random()*0.25)
    END AS descuento
  FROM lineas l
)
INSERT INTO fact_ventas
  (id_tiempo, id_producto, id_local, id_cliente, id_canal, id_medio_pago, id_promocion,
   ticket_id, unidades, importe_bruto, descuento, importe_neto, precio_unitario_neto)
SELECT
  m.id_tiempo, m.id_producto, m.id_local, m.id_cliente, m.id_canal,
  m.id_medio_pago, m.id_promocion,
```

```
m.ticket_id, m.unidades,
m.importe_bruto,
m.descuento,
m.importe_bruto - m.descuento AS importe_neto,
(m.importe_bruto - m.descuento) / NULLIF(m.unidades,0) AS precio_unitario_neto
from medidas m;
```

Queries útiles con modelo estrella:

- **AOV (ticket promedio) por canal:**

La consulta calcula el valor promedio por ticket (AOV) en cada canal de venta.
Para cada canal suma el importe neto de las ventas y lo divide por la cantidad de tickets distintos. El resultado permite comparar el gasto promedio de los clientes entre el canal físico y el canal ecommerce.

```
SELECT
   c.tipo_canal,
   ROUND(
     SUM(f.importe_neto)::numeric
     / NULLIF(COUNT(DISTINCT f.ticket_id), 0),
     2
   ) AS aov
FROM fact_ventas f
JOIN dim_canalventa c
  ON f.id_canal = c.id_canal
GROUP BY c.tipo_canal
ORDER BY aov DESC;
```

- **UPT (unidades por ticket) por mes:**

La consulta calcula el promedio de unidades por ticket (UPT) por año y mes.
Para cada mes, suma las unidades vendidas y las divide por la cantidad de tickets distintos, redondeando a 2 decimales. El resultado muestra cómo varía el UPT mes a mes, útil para analizar tendencias y estacionalidad.

```
SELECT
   t.anio,
   t.mes,
   ROUND(
     SUM(f.unidades)::numeric
     / NULLIF(COUNT(DISTINCT f.ticket_id), 0),
     2
   ) AS upt  -- unidades por ticket
FROM fact_ventas f
JOIN dim_tiempo t
  ON f.id_tiempo = t.id_tiempo
GROUP BY t.anio, t.mes
ORDER BY t.anio, t.mes;
```

- **AOV por canal y fin de semana**

La consulta calcula el valor promedio por ticket (AOV) diferenciando dos variables:

1. El **canal de venta** (físico o ecommerce).
2. Si la compra ocurrió en **fin de semana** o en **día hábil** (es_fin_de_semana).

Para cada combinación, suma el importe neto de las ventas y lo divide por la cantidad de tickets distintos, redondeado a 2 decimales.

El resultado permite comparar no solo qué canal tiene tickets de mayor valor, sino también si el comportamiento de compra varía entre fines de semana y días hábiles. Esto es útil para

entender patrones de consumo y planificar promociones específicas según el momento y el canal.

```
SELECT
  c.tipo_canal AS canal_venta,
  t.es_fin_de_semana,
  ROUND(
    SUM(v.importe_neto)::numeric / NULLIF(COUNT(DISTINCT
v.ticket_id), 0),
    2
  ) AS aov
FROM fact_ventas v
JOIN dim_canalventa c ON v.id_canal  = c.id_canal
JOIN dim_tiempo     t ON v.id_tiempo = t.id_tiempo
GROUP BY c.tipo_canal, t.es_fin_de_semana
ORDER BY c.tipo_canal, t.es_fin_de_semana;
```

- **UPT por categoría (promedio de unidades de esa categoría por ticket)**

La consulta calcula el promedio de unidades por ticket (UPT) para cada categoría de producto.
Para cada categoría, suma la cantidad total de unidades vendidas y la divide por el número de tickets distintos, redondeando el resultado a 2 decimales.

El objetivo es identificar en qué categorías los clientes suelen comprar más unidades por ticket, lo que permite detectar categorías con mayor capacidad de venta cruzada o con productos más comprados en conjunto.

```
SELECT
    p.categoria,
    ROUND(
      SUM(v.unidades)::numeric
      / NULLIF(COUNT(DISTINCT v.ticket_id), 0),
      2
    ) AS upt_categoria
FROM fact_ventas v
JOIN dim_producto p
  ON v.id_producto = p.id_producto
GROUP BY p.categoria
ORDER BY upt_categoria DESC;
```

Ingresos mensuales y crecimiento porcentual mes a mes (MoM)

Esta query calcula el **revenue total por mes** (suma de importe_neto de las ventas) y luego mide el **crecimiento porcentual respecto al mes anterior**.
En otras palabras, permite ver la evolución de las ventas mes a mes y detectar **tendencias, picos o caídas** en el ingreso.

Es útil para presentaciones de gestión porque muestra no solo los montos de venta, sino también **cómo varían** en el tiempo, resaltando periodos de crecimiento o retracción.

```
WITH m AS (
  SELECT t.anio, t.mes, SUM(f.importe_neto) AS revenue
  FROM fact_ventas f
  JOIN dim_tiempo t ON t.id_tiempo = f.id_tiempo
  GROUP BY t.anio, t.mes
)
SELECT
```

```
  anio, mes, revenue,

  ROUND( (revenue - LAG(revenue) OVER (ORDER BY anio, mes))

        / NULLIF(LAG(revenue) OVER (ORDER BY anio, mes),0)::numeric , 4) AS
mom_growth

FROM m

ORDER BY anio, mes;
```