

INFORME ACADÉMICO

Proyecto 1

AUTOR

Kevin Steven Ramirez Torres 2259371

Diego Alejandro Tolosa Sanchez

DOCENTE

Hector Fabio Ocampo Arbelaez

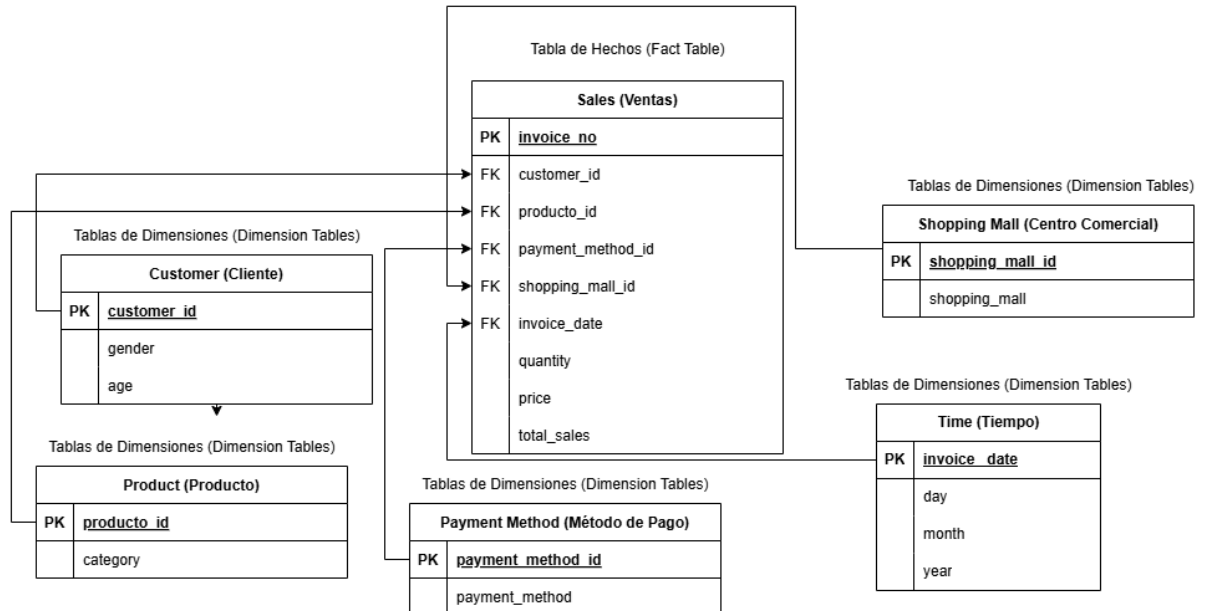
Universidad del Valle – Seccional Tuluá

Facultad de Ingeniería

Ingeniería de Sistemas

2025-1

1. Diseño del Modelo de la Bodega de Datos



justificación del modelo empleado

Elegimos usar el **Modelo Estrella** primero por su simplicidad y facilidad de implementación, pues creemos que este modelo es más simple que el de **Copo de Nieve**, ya que nuestro Diagrama de bases de datos consiste de una única tabla de hechos central. Además este modelo es intuitivo cuando el equipo no tiene un conocimiento limitado de los modelos de datos. También, se eligió por su eficiencia en las consultas ya que reduce la cantidad de joins para realizar consultas en la base de datos.

Script SQL de la creación de las tablas en PostgreSQL

```
-- Tabla de Cliente
CREATE TABLE customer (
    customer_id VARCHAR(50) PRIMARY KEY, -- Cambiado a VARCHAR
    gender VARCHAR(10),
    age INT
);

-- Tabla de Producto
CREATE TABLE product (
    product_id SERIAL PRIMARY KEY,
    category VARCHAR(50));
```

```
-- Tabla de Método de Pago
CREATE TABLE paymentmethod (
    payment_method_id SERIAL PRIMARY KEY,
    payment_method VARCHAR(50)
);

-- Tabla de Centro Comercial
CREATE TABLE shoppingmall (
    shopping_mall_id SERIAL PRIMARY KEY,
    shopping_mall VARCHAR(100)
);

-- Tabla de Tiempo
CREATE TABLE time (
    invoice_date DATE PRIMARY KEY,
    day INT,
    month INT,
    year INT
);

-- Tabla de Hechos (Ventas)
CREATE TABLE sales (
    invoice_no VARCHAR(20) PRIMARY KEY,
    customer_id VARCHAR(50) REFERENCES customer(customer_id),
    product_id INT REFERENCES product(product_id),
    payment_method_id INT REFERENCES
paymentmethod(payment_method_id),
    shopping_mall_id INT REFERENCES
shoppingmall(shopping_mall_id),
    invoice_date DATE REFERENCES time(invoice_date),
    quantity INT,
    price DECIMAL(10, 2),
    total_sales DECIMAL(10, 2)
);
```

2. Extracción, Transformación y Carga de Datos

Código del Proceso ETL en Python.

```
# Extracción: Cargar datos desde un archivo CSV
# -----
print("Extrayendo datos del CSV...")
csv_file = "customer_shopping_data.csv" # Rutas del archivo CSV
# Carga el dataset desde el archivo CSV
df = pd.read_csv(csv_file)
print("Datos Extraídos correctamente 👍")

# Mostrar las primeras filas para verificar la carga
print(df.head())
```

El primer paso es leer el archivo CSV usando la librería **Pandas** en Python.

- El archivo CSV se carga en un DataFrame de Pandas
- Podemos inspeccionar los datos con **df.head()** para asegurar de que se cargaron correctamente.

```
# Crear la tabla de dimensiones: Customer
customer_df = df[['customer_id', 'gender', 'age']].drop_duplicates().reset_index(drop=True)

# Crear la tabla de dimensiones: Product
product_df = df[['category']].drop_duplicates().reset_index(drop=True)
product_df['product_id'] = product_df.index + 1 # Asignar un ID a cada categoria

# Crear la tabla de dimensiones: PaymentMethod
payment_method_df = df[['payment_method']].drop_duplicates().reset_index(drop=True)
payment_method_df['payment_method_id'] = payment_method_df.index + 1 # Asignar un ID a cada metodo de pago

# Crear la tabla de dimensiones: ShoppingMall
shopping_mall_df = df[['shopping_mall']].drop_duplicates().reset_index(drop=True)
shopping_mall_df['shopping_mall_id'] = shopping_mall_df.index + 1 # Asignar un ID único a cada centro comercial

# Crear la tabla de dimensiones: Time
df['invoice_date'] = pd.to_datetime(df['invoice_date'], format='%d/%m/%Y') # Convertir a tipo fecha
time_df = df[['invoice_date']].drop_duplicates().reset_index(drop=True)
time_df['day'] = time_df['invoice_date'].dt.day
time_df['month'] = time_df['invoice_date'].dt.month
time_df['year'] = time_df['invoice_date'].dt.year
```

Los tratamientos de datos empleados para este proyecto fueron los siguientes:

eliminación de datos duplicados, reinicio de índices, creación y asignación de id y modificación de un tipo de dato, debido a que la tabla de time proporcionada por el dataset

tiene un formato que no es compatible con el formato datetime de postgres hicimos una conversión.

Cargar los datos en la bodega de datos en PostgreSQL

```
# Configurar la conexión a PostgreSQL
engine =
create_engine("postgresql://postgres:admin@localhost:5432/Prueba")

# Cargar las tablas de dimensiones
customer_df.to_sql("customer", engine, if_exists="replace",
index=False)
product_df.to_sql('product', engine, if_exists='replace', index=False)
payment_method_df.to_sql('paymentmethod', engine, if_exists='replace',
index=False)
shopping_mall_df.to_sql('shoppingmall', engine, if_exists='replace',
index=False)
time_df.to_sql('time', engine, if_exists='replace', index=False)

# Cargar la tabla de hechos
sales_df.to_sql('sales', engine, if_exists='replace', index=False)

print("Datos cargados correctamente en PostgreSQL 👍")
```

Una vez transformados los datos, el siguiente paso es cargarlos en la base de datos PostgreSQL, para esto puedes usar la librería **SQLAlchemy**, que facilita la conexión y la carga de datos.

- Se crea una conexión a la base de datos PostgreSQL usando SQLAlchemy.
- Se cargan los DataFrames en las tablas correspondientes usando **to_sql**.
- El parámetro **if_exists='replace'** asegura que los datos se agreguen a las tablas existentes sin borrar los datos previos.

Comprobación de que los datos han sido correctamente insertados en la base de datos.

	invoice_no text	customer_id text	product_id bigint	payment_method_id bigint	shopping_mall_id bigint	invoice_date timestamp without time zone	quantity bigint	price double precision	total_sales double precision
1	I138884	C241288	1	1	1	2022-08-05 00:00:00	5	1500.4	7502
2	I317333	C111565	2	2	2	2021-12-12 00:00:00	3	1800.51	5401.53
3	I127801	C266599	1	3	3	2021-11-09 00:00:00	1	300.08	300.08
4	I173702	C988172	2	1	4	2021-05-16 00:00:00	5	3000.85	15004.25
5	I337046	C189076	3	3	1	2021-10-24 00:00:00	4	60.6	242.4
6	I227836	C657758	1	1	2	2022-05-24 00:00:00	5	1500.4	7502
7	I121056	C151197	4	3	5	2022-03-13 00:00:00	1	40.66	40.66
Total rows: 99457 Query complete 00:00:01.422									

3. Consultas Analíticas en SQL

a. Total de ventas por Categoría de productos

```
SELECT
    p.category AS categoria,
    SUM(s.total_sales) AS total_ventas
FROM
    sales s
JOIN
    product p ON s.product_id = p.product_id
GROUP BY
    p.category
ORDER BY
    total_ventas DESC;
```



- Esta consulta calcula el total de ventas por categoría de producto.
- Se une la tabla **sales** con la tabla **product** para obtener la categoría de cada producto.
- Se agrupan los resultados por categoría y se ordenan de mayor a menor.

	categoria text	total_ventas double precision
1	Clothing	113996791.03997043
2	Shoes	66553451.470001996
3	Technology	57862350
4	Cosmetics	6792862.8999999957
5	Toys	3980426.2399998284
6	Food & Beverage	849535.0499999993
7	Books	834552.8999999999
Total rows: 8 Query complete 00:00:00.33		

b. Clientes con mayor volumen de compras

```
SELECT
    c.customer_id AS cliente,
    SUM(s.total_sales) AS total_compras
FROM
    sales s
JOIN
    customer c ON s.customer_id = c.customer_id
GROUP BY
    c.customer_id
ORDER BY
    total_compras DESC
LIMIT 10;
```



- Esta consulta identifica a los clientes que han realizado las mayores compras en términos de volumen.
- Se une la tabla **sales** con la tabla **customer** para obtener los datos del cliente.
- Se agrupan los resultados por cliente y se ordenan de mayor a menor.
- Se limita el resultado a los 10 clientes con mayor volumen de compras.

	cliente  text	total_compras  double precision
1	C922102	26250
2	C326518	26250
3	C292192	26250
4	C242994	26250
5	C233437	26250
6	C167779	26250
7	C509633	26250
Total rows: 10		Query complete C

c. Métodos de pago más utilizados

```
SELECT
    pm.payment_method AS metodo_pago,
    COUNT(*) AS total_transacciones
FROM
    sales s
JOIN
    paymentmethod pm ON s.payment_method_id = pm.payment_method_id
GROUP BY
    pm.payment_method
ORDER BY
    total_transacciones DESC;
```

- Esta consulta cuenta cuantas transacciones se han realizado con cada método de pago.
- Se une la tabla **sales** con la tabla **paymentmethod** para obtener el nombre del método de pago.
- Se agrupan los resultados por método de pago y se ordenan de mayor a menor.

	metodo_pago 	total_transacciones 
	text	bigint
1	Cash	44447
2	Credit Card	34931
3	Debit Card	20079

d. Comparación de ventas por mes

```
SELECT
    EXTRACT(YEAR FROM s.invoice_date) AS año,
    EXTRACT(MONTH FROM s.invoice_date) AS mes,
    SUM(s.total_sales) AS total_ventas
FROM
    sales s
GROUP BY
    año, mes
ORDER BY
    año, mes;
```

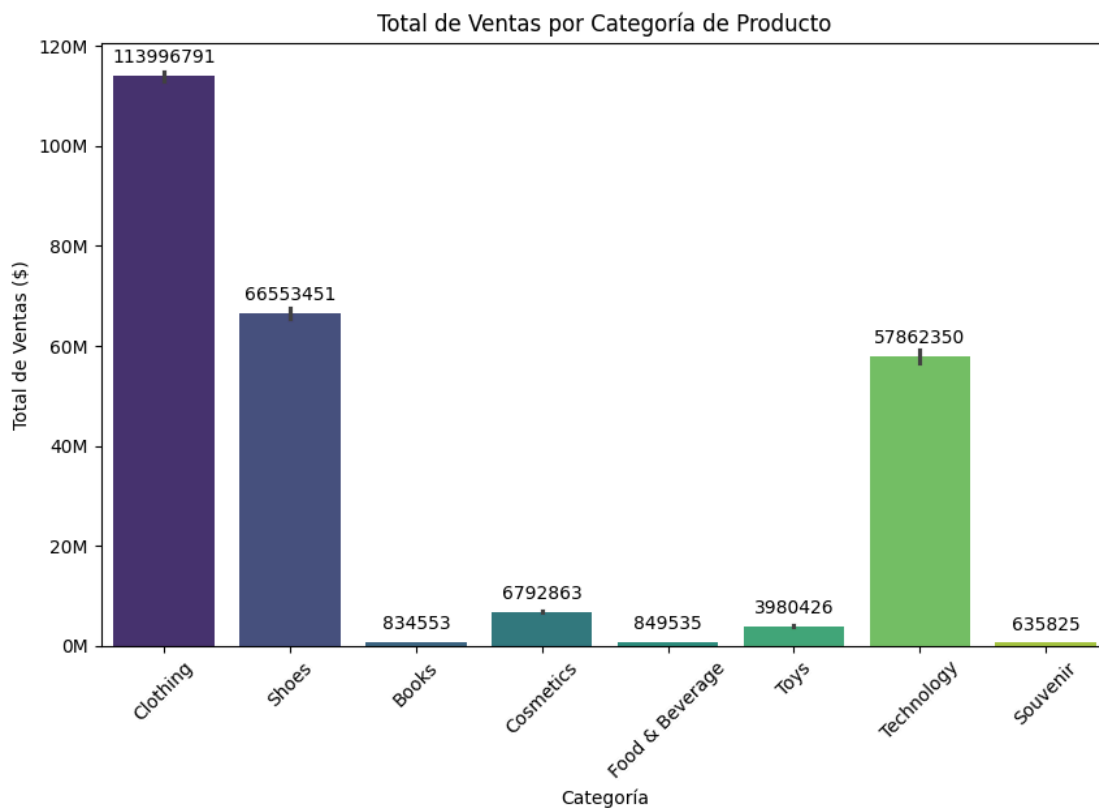
- Esta consulta calcula el total de ventas por mes y año.

- Se utiliza la función **Extract** para obtener el año y el mes de la fecha de la factura.
- Se agrupan los resultados por año y mes, y se ordenan cronológicamente.

	año numeric	mes numeric	total_ventas double precision
1	2021	1	9641614.620000066
2	2021	2	8772315.220000051
3	2021	3	9455359.380000075
4	2021	4	9389541.540000008
5	2021	5	9771756.970000006
6	2021	6	9286271.350000052
7	2021	7	10311119.68000001
Total rows: 27		Query complete 00:00:00.516	

4. Análisis Descriptivo y Visualización de Datos

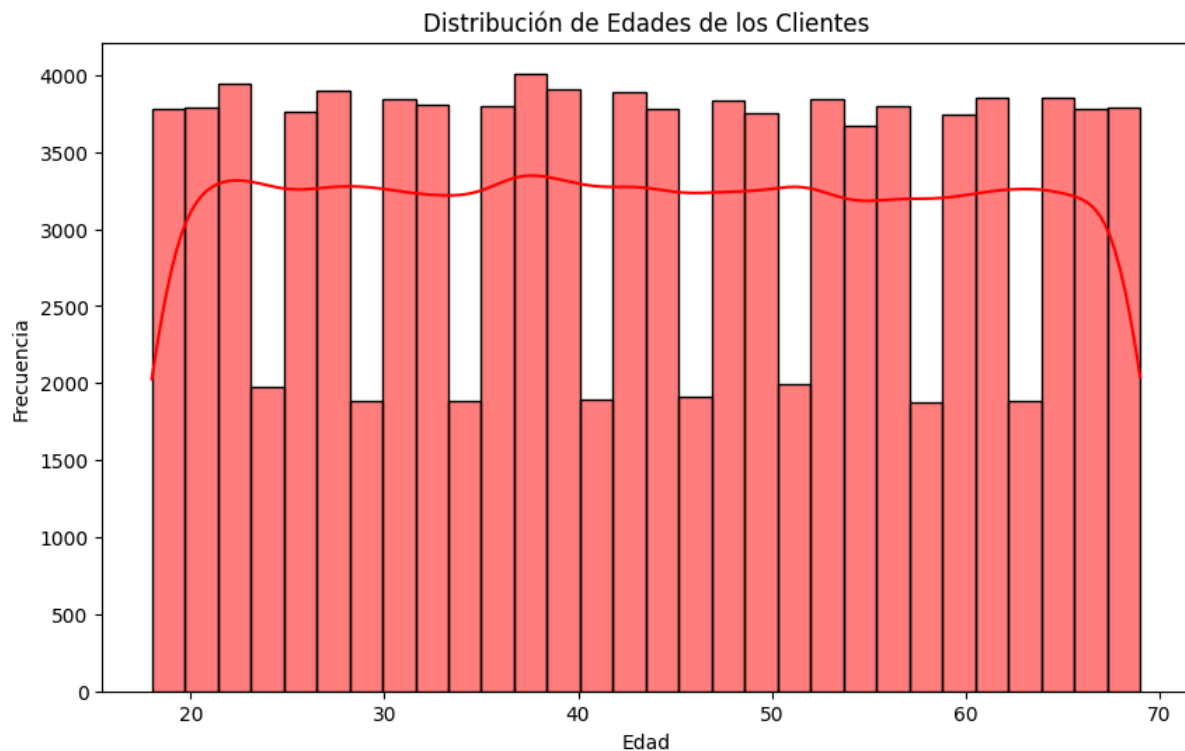
- **Total de ventas por categoría de producto**



Identificamos que las ventas están concentradas en unas pocas categorías, como “Clothing”, “Shoes” y “Technology”. Las categorías más rentables son clave para el negocio,

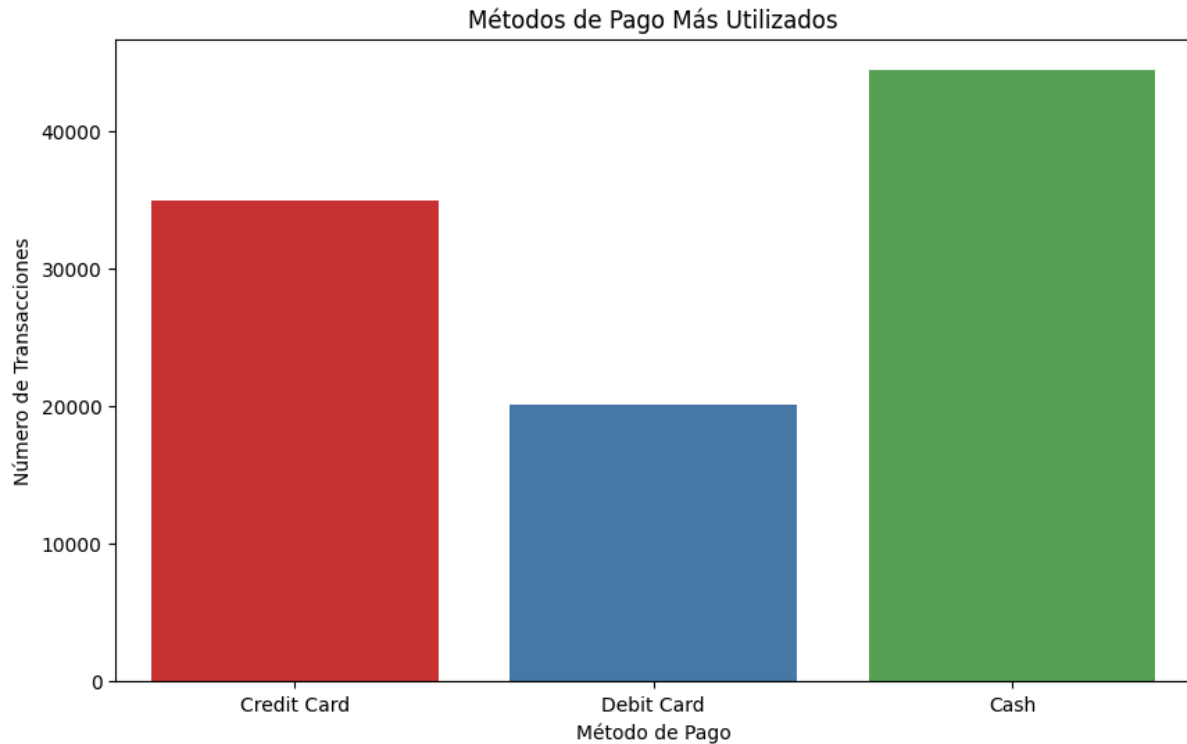
mientras que las categorías con bajas ventas necesitan atención. Con lo que podríamos enfocar los recursos en las categorías más rentables o mejorar el desempeño de las categorías con bajas ventas.

- **Distribución de las edad de los Clientes**



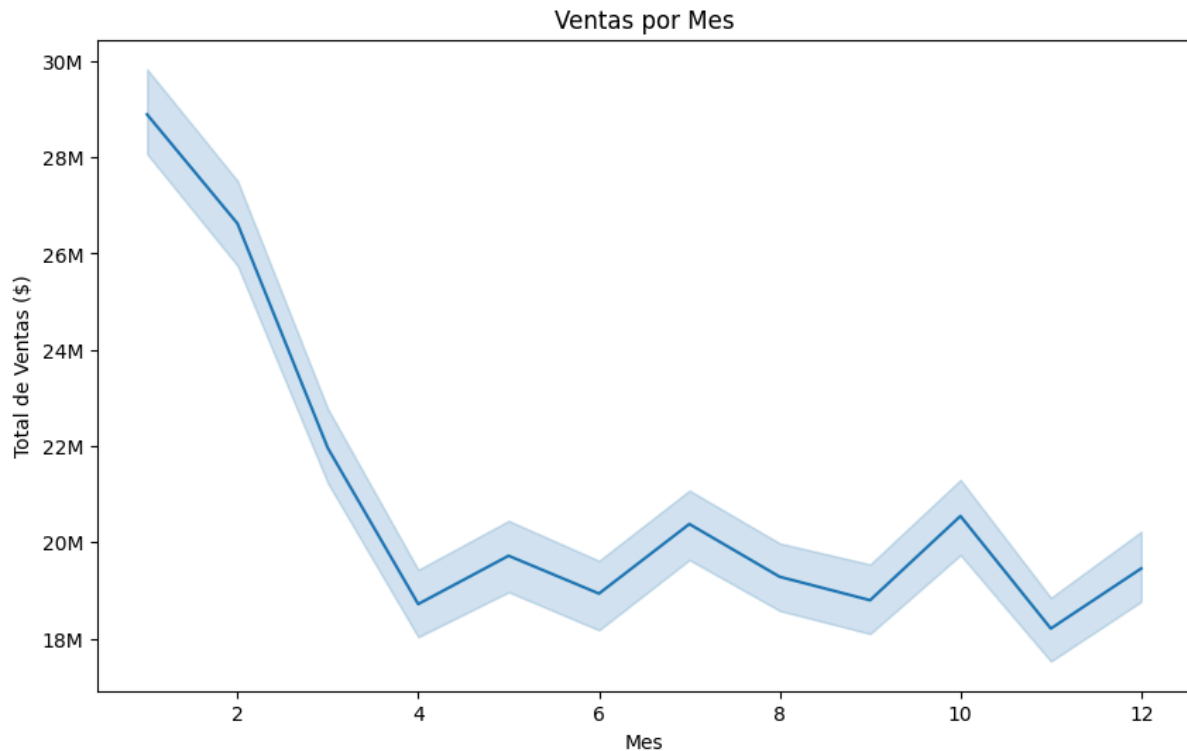
Mediante este gráfico podemos conocer los rangos de edades que realizan más compras, nos permite centrarnos en un público objetivo

- **Métodos de pago más utilizados**



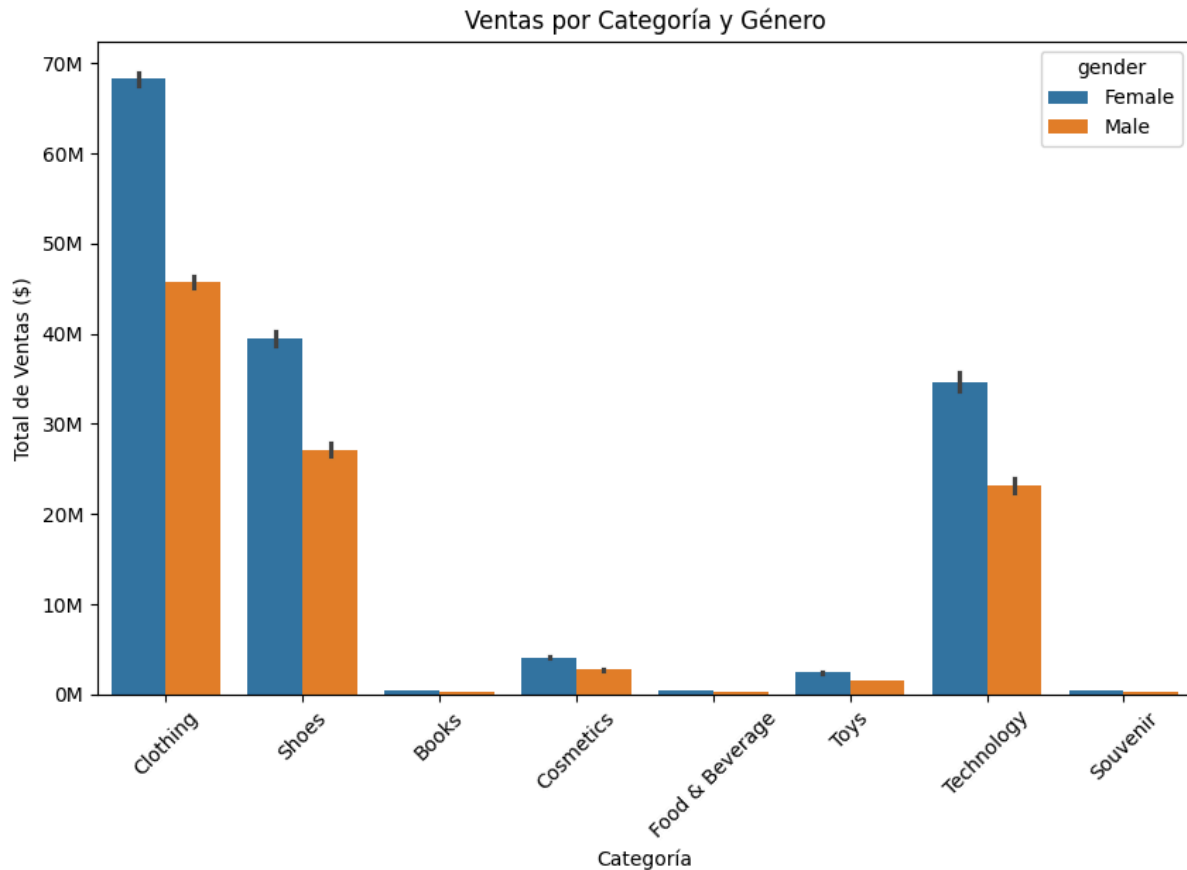
Podemos ver que el método de pago más utilizado por los clientes es el **Efectivo**, luego sigue la **Credit Card** y por último la **Debit Card**. Lo que nos indica una clara preferencia por el pago en **efectivo**. Con estos datos podemos realizar un análisis para entender las razones por las que las personas prefieren ciertos métodos de pago y buscar optimizar los menos utilizados.

- Ventas por Mes



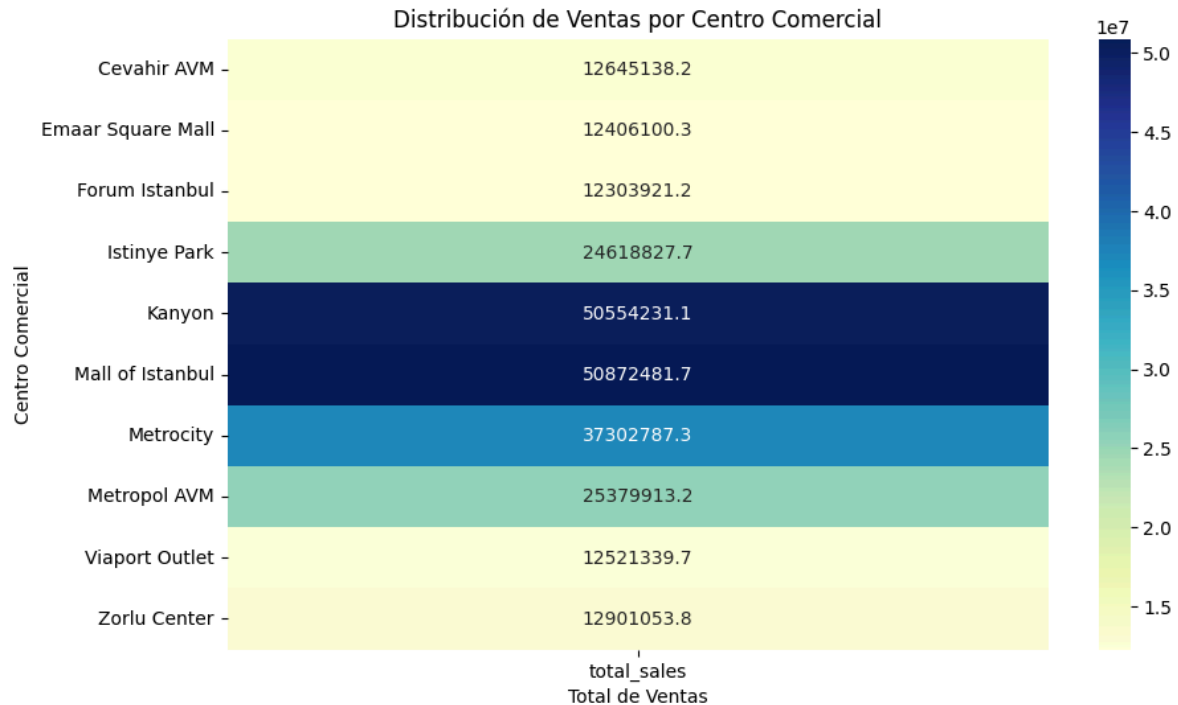
El gráfico nos permite ver que tenemos un pico alto el primer mes, seguido de una caída hasta el mes 4, con ventas bajas y fluctuaciones en los meses intermedio del año. Conocer esta información nos permite implementar o buscar estrategias para mejorar el desempeño en los meses bajos tratando de mantener la demanda y ver qué productos fueron exitosos en los meses fuertes para replicar las estrategias en otros periodos.

- Ventas por Categoría y Género



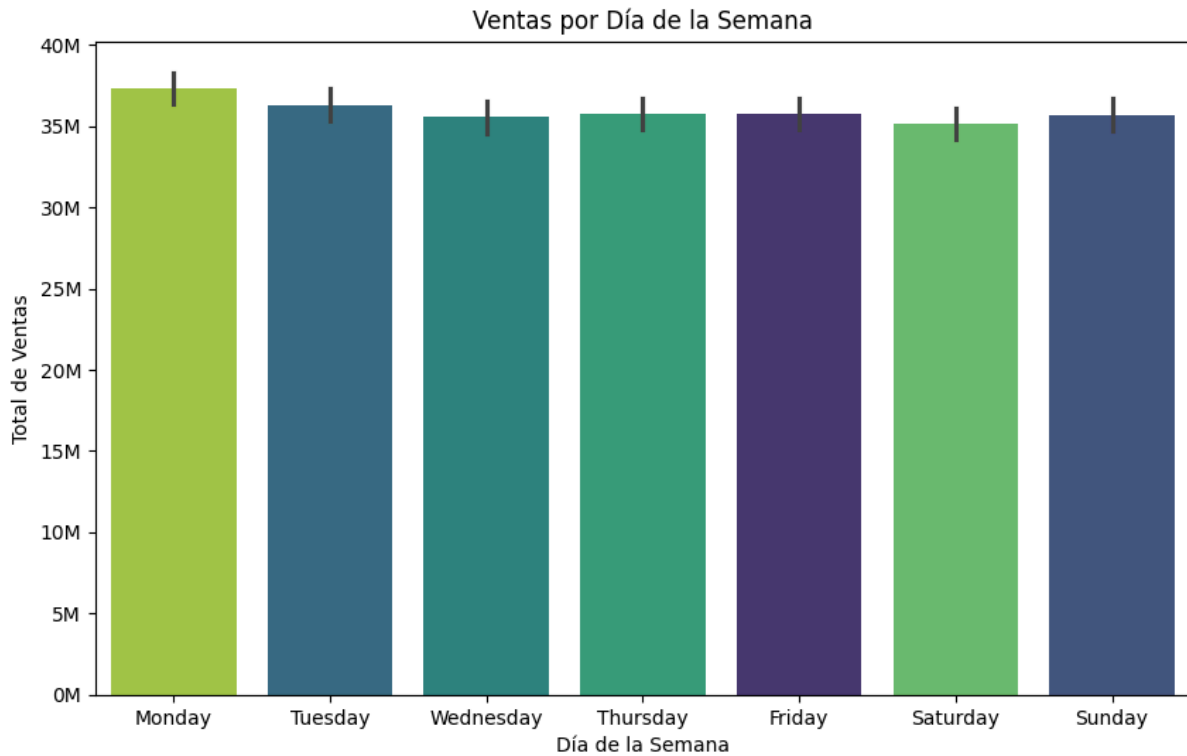
El gráfico nos permite ver qué categorías dominan en ventas, además ver que las mujeres tienen una gran participación en esas ventas y categorías. Por otra parte los hombres tienen presencia en la categoría tecnología. Las demás categorías tienen ventas bajas, lo que sugiere revisar o implementar estrategias que permitan aumentar sus ventas.

- **Distribución de ventas por centro comercial**



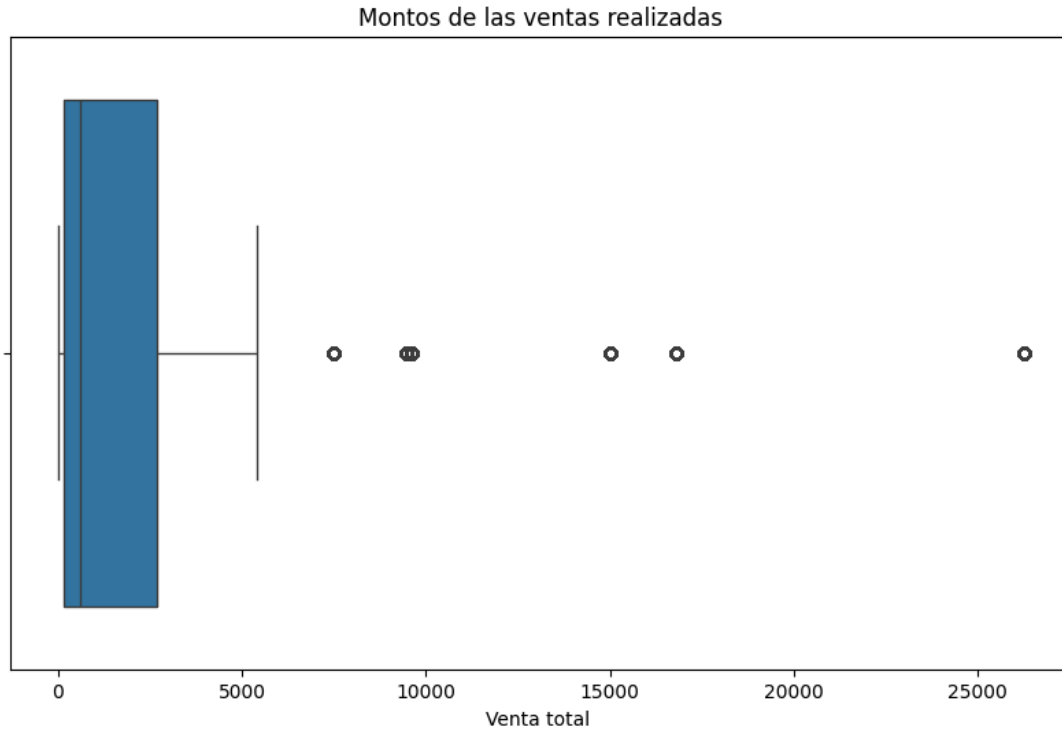
Este gráfico nos permite tener una información detallada sobre en cuál de los centros comerciales se realizaron más ventas, mediante un mapa de calor. Esta gráfica es muy importante para conocer a futuro otro tipo de datos, como podría ser la ubicación de un siguiente centro comercial o en qué lugar podría ser más eficiente colocar otro centro comercial.

- Ventas por día de la semana



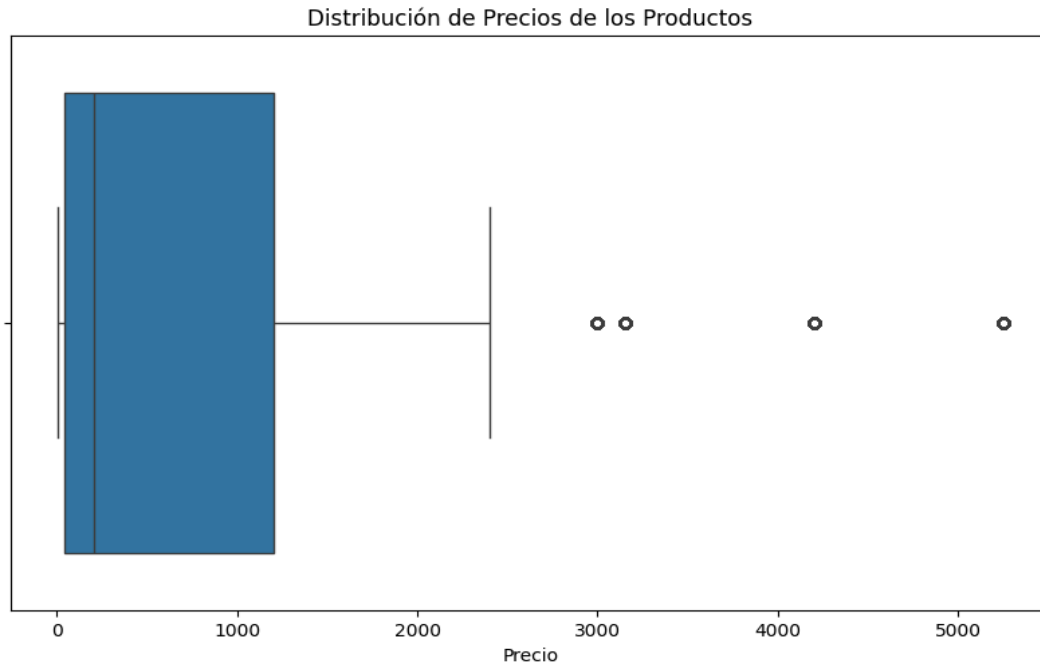
Este gráfico nos permite tener una información detallada sobre las ventas por día de la semana lo que nos permite saber que días son los que más ventas tiene y analizar porque tiene más ventas esos días para replicarlos en los otros días en los que no se tengan buenas ventas. Pero en este caso todos los días tiene muy buenas ventas pero igual se puede mejorar.

- **Visualización de los montos de las compras hechas**



Este gráfico de caja nos muestra donde se agrupan los montos de las ventas hechas por los clientes, se puede evidenciar que la gran mayoría de compras son entre los 0 y 5000, aunque hay ventas que se van hasta más de los 25000.

- **Distribución de precios de los productos**



Este gráfico muestra en qué precios se agrupan la mayoría de los productos y a su vez que algunos precios de los productos son mucho más elevados y este gráfico nos permite conocerlos.

- **Posibles mejoras para el negocio**

Incrementar el stock en las categorías más vendidas: Dado que “Clothing”, “Shoes” y “Technology” son las categorías más rentables sobre todo para las mujeres las cuales hicieron más compras

Incentivar el uso de tarjetas de crédito y débito como el pago en efectivo es el más usado, se pueden ofrecer descuentos o recompensas por el uso de tarjetas para mejorar la seguridad y control financiero.

Apertura de nuevas tiendas o ampliaciones basado en los datos de ventas por ubicación, se pueden tomar decisiones sobre la expansión en los centros comerciales más rentables o buscar zonas similares.

- **Conclusiones del Proyecto**

Eficiencia del Modelo de Datos

La elección del modelo estrella resultó adecuada para la bodega de datos debido a su simplicidad, facilidad de implementación y rapidez en las consultas. La estructura optimizada con una tabla de hechos central y tablas de dimensiones permite una mejor organización de los datos y una ejecución eficiente de análisis.

Los gráficos y análisis de tendencias revelaron patrones importantes, como picos de ventas, caídas en ciertos meses y distribución de compras por rangos de precios.

La visualización con mapas de calor y diagramas de caja facilitó la interpretación de la información para definir estrategias de negocio.