# 📝INFORME FINAL DEL PROYECTO ECOMMERCE

## 🔎Introducción

El objetivo de este proyecto fue realizar un **análisis integral de un e-commerce** utilizando un enfoque completo de *ETL + Análisis + Visualización*. El proceso se dividió en dos fases principales:

1. **Limpieza, transformación y consolidación de datos** mediante *Python (Pandas)*.
2. **Desarrollo de dashboards analíticos** en *Power BI* con KPIs financieros, de clientes y logísticos.

Los datasets provienen de diferentes fuentes que representan información de pedidos, productos, clientes y pagos.

---

## 📂Archivos de datos utilizados

| Archivo | Descripción |
|---|---|
| **Ordenes** | Dataset principal con información de pedidos, fechas y estados. |
| **df_Customers** | Datos de clientes: ID, ciudad y estado. |
| **df_Payments** | Detalles de pagos: tipo de pago, cuotas e importes. |
| **df_Products** | Datos de productos y categorías. |

Se integraron todos los datasets en un modelo final denominado `fact_table.xlsx`, que se utilizó para el modelado y análisis en Power BI.

---

## 🛠️Fase 1: Limpieza y transformación (Python / Pandas)

### 1.1 Exploración inicial

- Se inspeccionaron los datasets para detectar valores nulos, tipos de datos inconsistentes y duplicados.
- Se identificó la necesidad de unificar todos los archivos mediante *joins* sobre las columnas `order_id` y `customer_id`.

### 1.2 Eliminación de duplicados

- Se eliminaron registros repetidos en `customer_id` y `order_id` para evitar duplicidades en ventas.
- Los `seller_id` duplicados se mantuvieron ya que pertenecían a líneas distintas de la misma orden.

### 1.3 Tratamiento de valores nulos

- Se eliminaron 1800 registros con `customer_id` o `product_category_name` vacíos.
- Se sustituyeron valores nulos en importes (`price`, `freight_value`) por 0, tras comprobar que representaban cancelaciones.

### 1.4 Homogeneización de tipos de datos

- Las columnas de fechas (`order_purchase_timestamp`, `order_delivered_customer_date`, etc.) se transformaron al tipo *datetime*.
- Los importes (`price`, `payment_value`) se convirtieron a *float*.
- Los identificadores (`order_id`, `product_id`, `customer_id`, `seller_id`) se establecieron como *string*.

### 1.5 Creación de nuevas columnas

Se agregaron columnas derivadas para enriquecer el análisis:

| Columna | Descripción | Motivo |
|---|---|---|
| `purchase_date`, `purchase_month`, `purchase_year` | Fechas derivadas de compra | Análisis temporal |
| `hours_to_approve`, `days_to_deliver` | Diferencia entre fechas clave | Medir eficiencia logística |
| `line_total` | Precio total de cada línea | Métrica de ingresos |
| `on_time_flag` | 1 si la entrega fue a tiempo, 0 si fue tardía | Evaluación logística |
| `delivered_flag` | 1 si fue entregado, 0 si no | Seguimiento de cumplimiento |

### 1.6 Filtrado final

- Se eliminaron filas con tiempos negativos o sin fecha de entrega válida.
- Se verificó coherencia entre fechas de compra, aprobación y entrega.

### 1.7 Resultado final

Tras aplicar todas las transformaciones, se consolidó el dataset final `fact_table.xlsx`, con 15 columnas limpias, listas para el modelado en Power BI.

---

## 📈 Fase 2: Modelado y análisis (Power BI)

### 2.1 Preparación en Power Query

- Se promovieron encabezados y ajustaron tipos de datos.
- Se eliminaron filas con `customer_id` o `product_category_name` vacíos.
- Se crearon columnas adicionales: `price_amt`, `shipping_amt`, `line_total_amt`, `approve_hours`, `delivery_hours`, `leadtime_est_days`.
- Se convirtieron los valores monetarios de centavos a euros.

### 2.2 Limpieza final en Power BI

Se eliminaron columnas obsoletas:

- `price`, `shipping_charges`, `line_total` (reemplazadas por versiones limpias).
- Se filtraron 5 filas con `on_time_flag_clean = null`.

---

## 🕐 Fase 3: Creación de medidas DAX

Las medidas se agruparon por área funcional:

### Ventas

- `Ingresos €` = SUM(line_total_amt)
- `Pedidos` = COUNT(order_id)
- `Ticket Medio €` = DIVIDE([Ingresos €], [Pedidos])
- `Unidades` = COUNT(product_id)

### Clientes

- `Clientes Únicos` = DISTINCTCOUNT(customer_id)
- `Pedidos por Cliente` = DIVIDE([Pedidos], [Clientes Únicos])
- `Ingresos por Cliente €` = DIVIDE([Ingresos €], [Clientes Únicos])
- `Clientes Acumulados` = Cumulative DISTINCTCOUNT(customer_id)

### Logística

- `% Entregados` = DIVIDE([Pedidos Entregados], [Pedidos])
- `% On-Time` = DIVIDE([Pedidos On-Time], [Pedidos Entregados])
- `Horas Aprobación Promedio` = AVERAGE(hours_to_approve)
- `Horas Entrega Promedio` = AVERAGE(delivery_hours)
- `Días Retraso Promedio` = Promedio de días tardíos

---

## 📉 Fase 4: Dashboards

### 📬 Ventas

**Objetivo:** analizar ingresos, pedidos y rendimiento por categoría y región.

**KPIs:** Ingresos Totales, Pedidos, Ticket Medio, Unidades.
**Gráficos:**

- Línea: Ingresos € por mes.
- Barras: Ingresos € por estado.
- Barras: Ingresos € por categoría.
- Tabla: Top productos por ingresos.

---

### 📬Clientes

**Objetivo:** comprender el comportamiento y la distribución de los clientes.

**KPIs:** Clientes Únicos, Pedidos por Cliente, Ingresos por Cliente €. **Gráficos:**

- Línea: Clientes únicos por mes.
- Barras: Clientes por estado.
- Donut: Clientes por método de pago.
- Línea: Clientes acumulados.

---

### 🖱️Logística

**Objetivo:** medir eficiencia de entregas y tiempos logísticos.

**KPIs:** % On-Time, % Entregados, Horas Aprobación y Entrega Promedio. **Gráficos:**

- Línea: Pedidos entregados por mes.
- Barras: % On-Time por estado.
- Donut: Pedidos a tiempo vs tardíos.
- Tabla: Promedio de retraso por estado.

---

### 💴Resumen Ejecutivo (Portada)

**KPIs:** Ingresos Totales (€), Clientes Únicos, % On-Time. **Gráficos:**

- Pie: Ingresos € por categoría.
- Donut: Clientes por método de pago.
- Pie: Entregas a tiempo vs tardías.

**Autor:** Claudio Baldini – ThePower Business School (Octubre 2025)

---

## 📊Conclusiones

- El dataset refleja un modelo con **clientes únicos por pedido**, **sin recurrencia**.
  Esto se debe a que los **IDs de clientes, pedidos y productos son únicos**, por lo que no existen relaciones de repetición entre ellos.
- La categoría **Toys** concentra el **94% de las ventas** → *outlier positivo*.
- El 93,6% de pedidos se entregaron a tiempo → eficiencia logística alta.
- SP y RJ son los estados con mayor actividad comercial.
- La tarjeta de crédito representa el método de pago dominante (~70%).

**Insight final:** el negocio presenta un buen desempeño operativo, pero una **alta dependencia de una sola categoría** y **baja fidelización de clientes**, debido a la estructura del dataset (IDs únicos).

---

## 🗂️Estructura del repositorio

```
📂 Ecommerce_Analysis/
│
├── data_raw/                # Datos originales (df_Customers, df_Payments,
df_Products, Ordenes)
├── data_clean/              # Dataset final limpio (fact_table.xlsx)
├── notebooks/               # Scripts Python (limpieza y unión de tablas)
├── powerbi/                 # Dashboard (.pbix)
├── docs/                    # Informe y visuales exportados
└── README.md
```

---

## Conclusión general

El proyecto cumple con todos los requisitos del análisis de datos completo: integración, transformación, modelado, visualización e interpretación. El resultado final entrega una visión holística del e-commerce desde tres perspectivas: **financiera, de cliente y logística**, evidenciando un modelo de **ventas únicas por cliente y pedido**, con excelente eficiencia operativa pero oportunidades de diversificación.