

Galileo
UNIVERSIDAD
La Revolución en la Educación

Proyecto Final Ciencia de Datos en Python

Postgrado en Análisis y Predicción de Datos

SAUL ALEXANDER LÓPEZ CONTRERAS 24000138

JOSIAS EMANUEL CHOCOOJ AZMITIA 24000705

ERICK MANUEL ANDRADE ZUÑIGA 24000165

Introducción

Desarrollo de sistema de ingeniería de datos, empleando:

- AWS, desarrollo infraestructura transaccional
- SQL, proceso de ETL, para carga de datos
- Python, principal sistema de programación para desarrollo de los insumos del proyecto



Objetivos

General

- Desarrollo de sistema transaccional, integrando Python, AWS y SQL.

Específicos

- Implementar un sistema transaccional automatizado en AWS.
- Crear y poblar una base de datos utilizando Python.
- Diseñar y construir un data warehouse que responda a preguntas de negocio específicas.
- Desarrollar un proceso ETL para integrar, transformar y cargar datos.
- Realizar análisis de datos y visualización para responder a preguntas de negocio.

Infraestructura

Desarrollada en AWS

- Amazon RDS
- Amazon EC2

Base de datos

- Transaccional: MySQL
- DataWarehouse: PostgreSQL
- Carga de datos: Faker

ETL

- Carga de dimensiones: pandas

Análisis de preguntas de negocio

- Carga de dimensiones: pandas, matplotlib, seaborn



RDS



EC2

Código

Crear Bases de datos Transaccional y DataWarehouse

```
# Crear la base de datos MySQL de transacciones
```

```
try:
    response = aws_rds_conn.create_db_instance(
        DBInstanceIdentifier=config.get('TRANSACC', 'DB_INSTANCE_ID'),
        DBName=config.get('TRANSACC', 'DB_NAME'),
        DBInstanceClass='db.t2.micro',
        Engine='mysql',
        MasterUsername=config.get('TRANSACC', 'DB_USER'),
        MasterUserPassword=config.get('TRANSACC', 'DB_PASSWORD'),
        Port=int(config.get('TRANSACC', 'DB_PORT')),
        PubliclyAccessible=True,
        VpcSecurityGroupIds=[config.get('VPC', 'SECURITY_GROUP')],
        AllocatedStorage=20
    )
    print(response)
except aws_rds_conn.exceptions.DBInstanceAlreadyExistsFault as ex:
    print("La instancia de base de datos MySQL ya existe.")
except Exception as ex:
    print("Error al crear la base de datos MySQL:", ex)
```

```
# Crear la base de datos PostgreSQL para DWH
```

```
try:
    response = aws_rds_conn.create_db_instance(
        DBInstanceIdentifier=config.get('DWH', 'DB_INSTANCE_ID'),
        DBName=config.get('DWH', 'DB_NAME'),
        DBInstanceClass='db.t2.micro',
        Engine='postgres',
        MasterUsername=config.get('DWH', 'DB_USER'),
        MasterUserPassword=config.get('DWH', 'DB_PASSWORD'),
        Port=int(config.get('DWH', 'DB_PORT')),
        PubliclyAccessible=True,
        VpcSecurityGroupIds=[config.get('VPC', 'SECURITY_GROUP')],
        AllocatedStorage=20
    )
    print(response)
except aws_rds_conn.exceptions.DBInstanceAlreadyExistsFault as ex:
    print("La instancia de base de datos PostgreSQL ya existe.")
except Exception as ex:
    print("Error al crear la base de datos PostgreSQL:", ex)
```

Ingesta de datos

```
from sqlalchemy import create_engine, Column, Integer, String, DECIMAL, DateTime, ForeignKey
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker, relationship
from faker import Faker
import random

db_url = 'mysql+pymysql://root:password@localhost/transactional_db'
engine = create_engine(db_url)
Session = sessionmaker(bind=engine)
session = Session()
Base = declarative_base()

fake = Faker()
```

Crear dimensiones

```
from sqlalchemy import create_engine, Column, Integer, String, Numeric, DateTime, ForeignKey
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker, relationship

db_url = 'postgresql://usuario:contraseña@localhost/dwh_db'
engine = create_engine(db_url)
Session = sessionmaker(bind=engine)
session = Session()
Base = declarative_base()

# Tabla de dimensión para Categoría (SCD Tipo 1)
class DimCategoria(Base):
    __tablename__ = 'dim_categoria'
    idcategoria = Column(Integer, primary_key=True)
    nombre = Column(String(50))
    descripcion = Column(String(255))
    estado = Column(Integer)
    fecha_inicio = Column(DateTime)
    fecha_fin = Column(DateTime)

    def __repr__(self):
        return f"<DimCategoria(idcategoria={self.idcategoria}, nombre='{self.nombre}')>"
```


Proceso ETL

Extracción, transformación y carga de la dimensión Categoría

```
def etl_dim_categoria():
```

Extracción

```
df_categoria = pd.read_sql('SELECT * FROM categoria', transactional_engine)
```

Transformación

```
df_categoria_dim = df_categoria.rename(columns={'idcategoria': 'idcategoria', 'nombre': 'nombre',  
'descripcion': 'descripcion', 'estado': 'estado'})
```

```
df_categoria_dim['fecha_inicio'] = pd.to_datetime('today')
```

```
df_categoria_dim['fecha_fin'] = pd.to_datetime('2999-12-31')
```

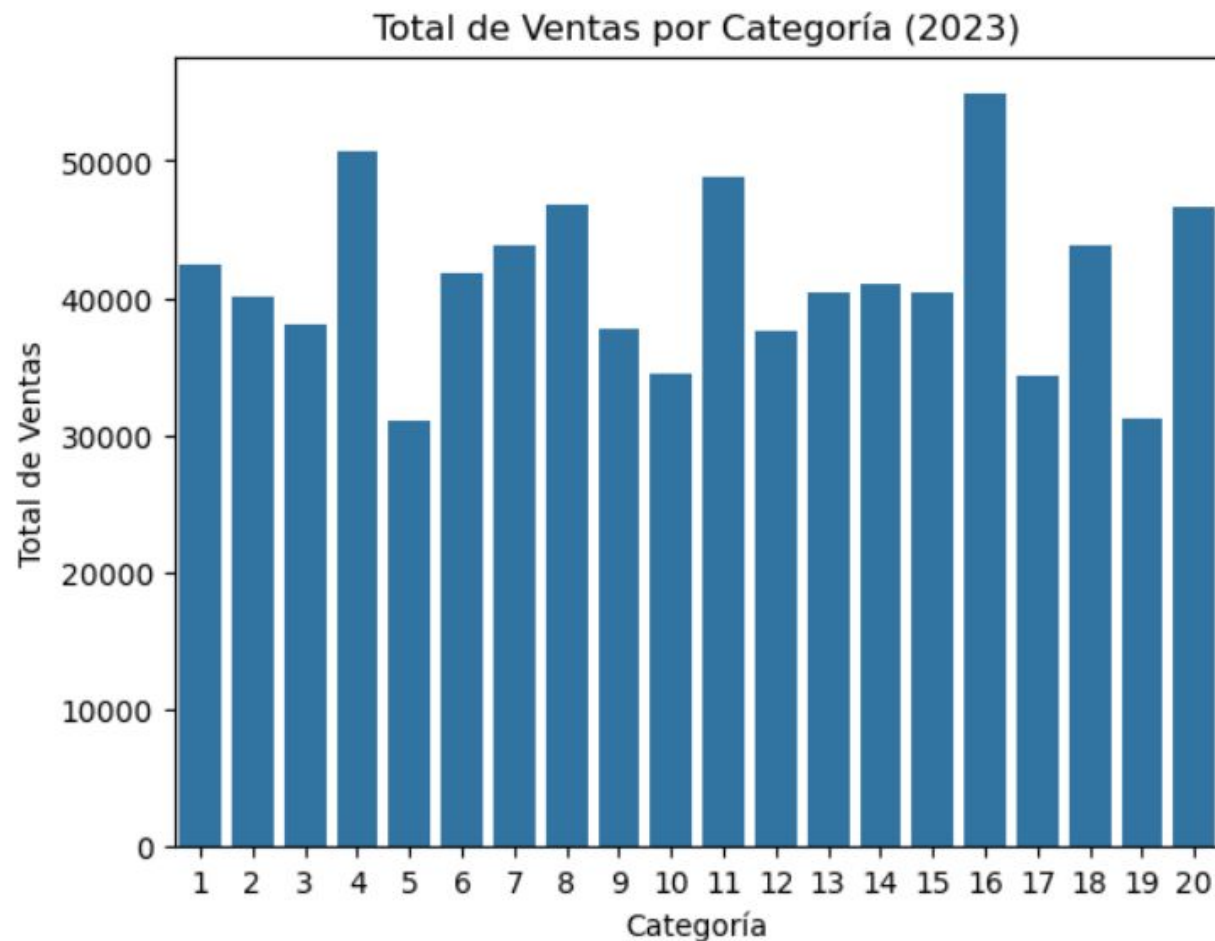
Carga

```
df_categoria_dim.to_sql('dim_categoria', dwh_engine, if_exists='append', index=False)
```

Preguntas de negocio

¿Cuál es el total de ventas por categoría de artículo en un período específico?

Comparativa de las ventas totales, por categoría de artículo en 2023.

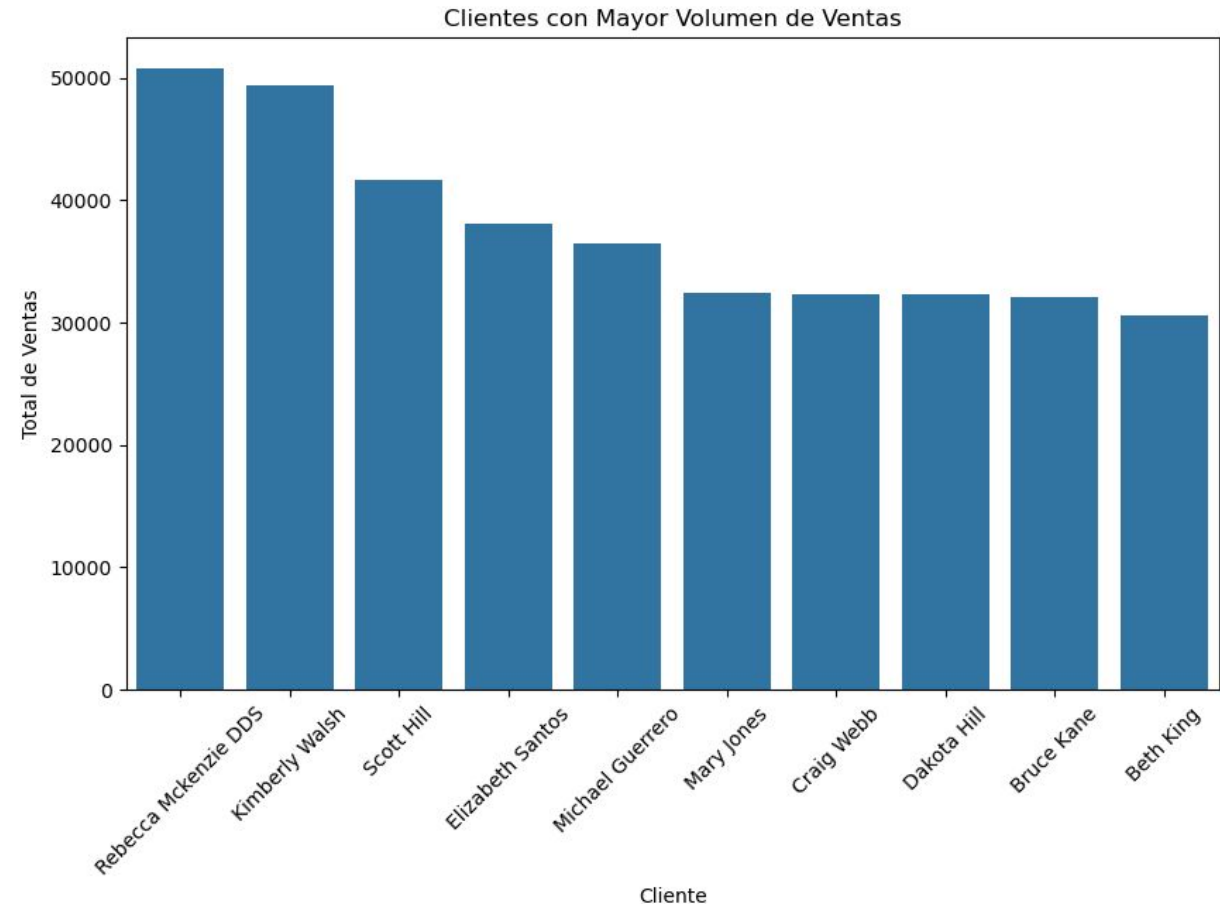


Código visualización

```
sns.barplot(x='idcategoria', y='precio', data=ventas_por_categoria)
plt.xlabel('Categoría')
plt.ylabel('Total de Ventas')
plt.title('Total de Ventas por Categoría (2023)')
plt.show()
```

¿Quiénes son los clientes que generan el mayor volumen de ventas y cuál es su contribución al total de ingresos?

Comparativa de clientes de acuerdo con cantidad de ventas que se les realizaron, orden descendente.

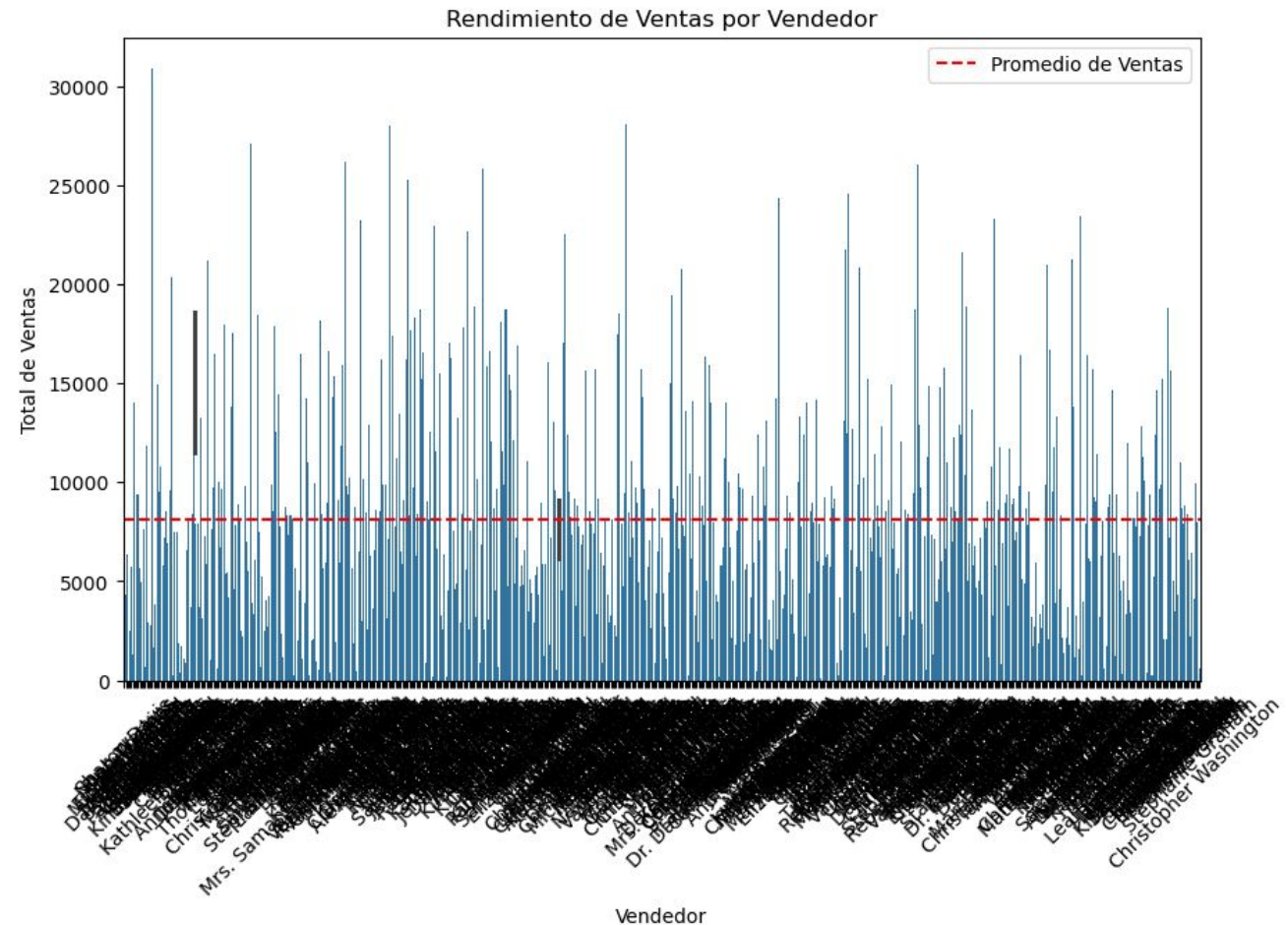


Código visualización

```
plt.figure(figsize=(10, 6))
sns.barplot(x='nombre', y='total', data=ventas_por_cliente.head(10))
plt.xlabel('Cliente')
plt.ylabel('Total de Ventas')
plt.title('Clientes con Mayor Volumen de Ventas')
plt.xticks(rotation=45)
plt.show()
```

¿Cuál es el rendimiento de ventas de cada vendedor y cómo se compara con el promedio de ventas por vendedor?

Comparativa de vendedores totales, por ventas obtenidas. Se identifica promedio de ventas, para destacar vendedores por encima del promedio.

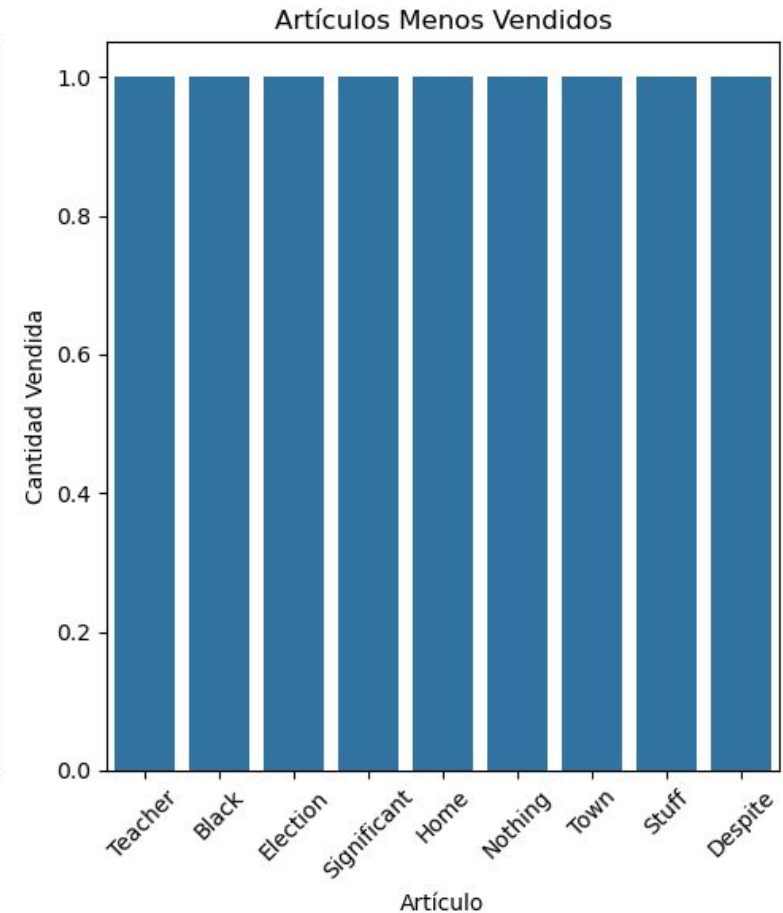
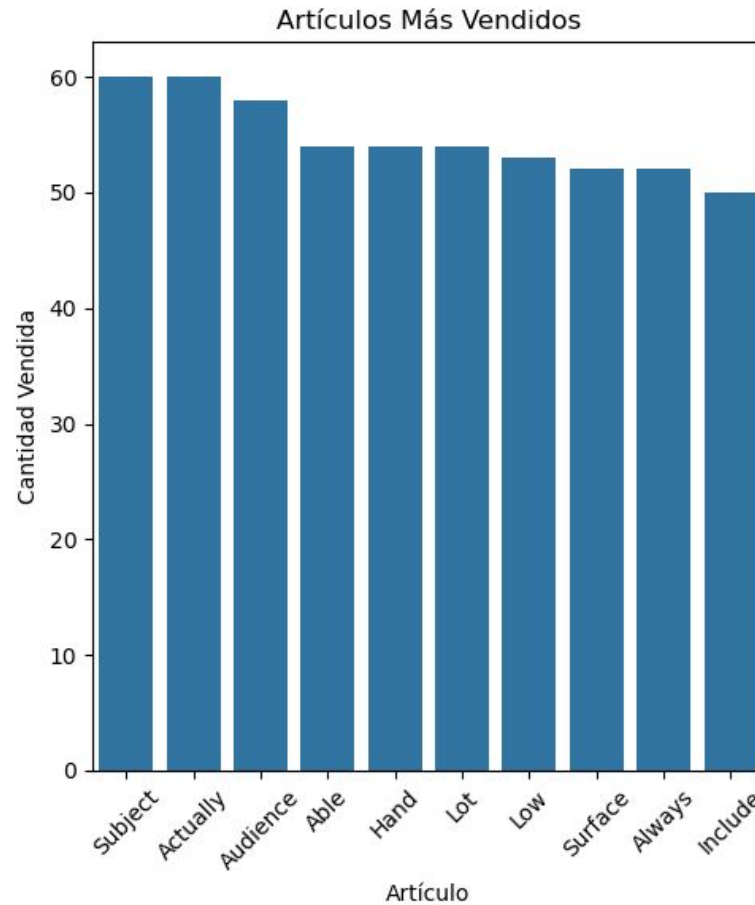


Código visualización

```
plt.figure(figsize=(10, 6))
sns.barplot(x='nombre', y='total', data=ventas_por_vendedor)
plt.axhline(promedio_ventas, color='red', linestyle='--', label='Promedio de Ventas')
plt.xlabel('Vendedor')
plt.ylabel('Total de Ventas')
plt.title('Rendimiento de Ventas por Vendedor')
plt.xticks(rotation=45)
plt.legend()
plt.show()
```


¿Cuáles son los artículos más vendidos y menos vendidos en términos de cantidad y monto total de ventas?

Detalle de artículos con mayor y menor cantidad de ventas.



Código visualización

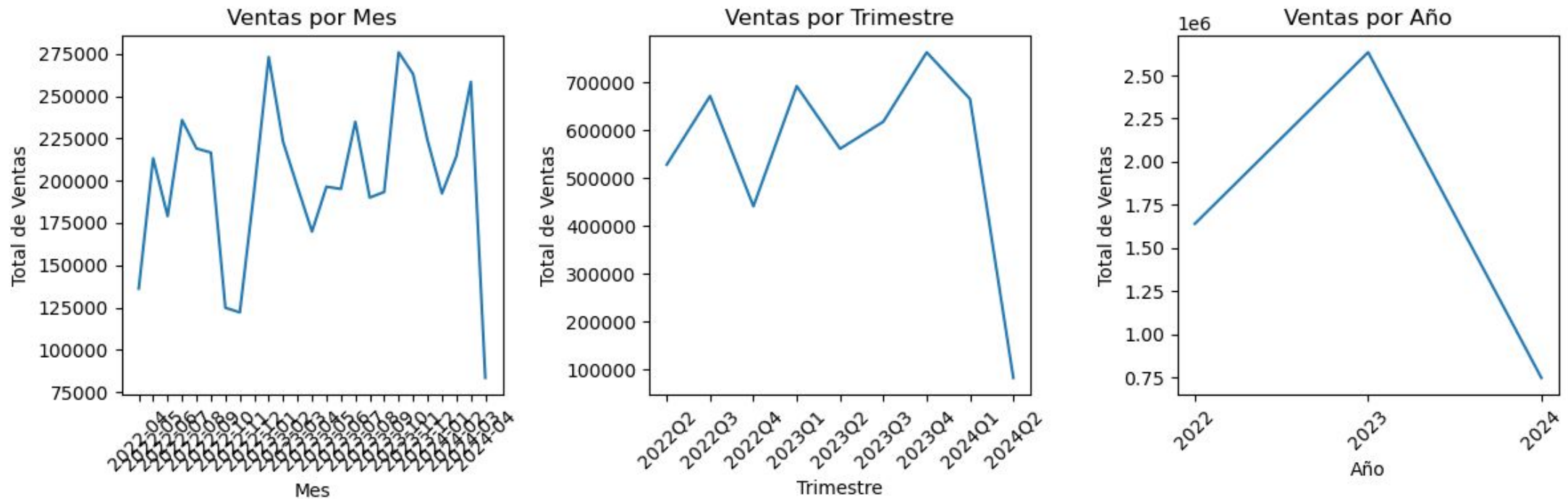
```
plt.figure(figsize=(10, 6))
plt.subplot(1, 2, 1)
sns.barplot(x='nombre', y='cantidad', data=articulos_mas_vendidos)
plt.xlabel('Artículo')
plt.ylabel('Cantidad Vendida')
plt.title('Artículos Más Vendidos')
plt.xticks(rotation=45)

plt.subplot(1, 2, 2)
sns.barplot(x='nombre', y='cantidad', data=articulos_menos_vendidos)
plt.xlabel('Artículo')
plt.ylabel('Cantidad Vendida')
plt.title('Artículos Menos Vendidos')
plt.xticks(rotation=45)

plt.tight_layout()
plt.show()
```

¿Cómo ha evolucionado el volumen de ventas a lo largo del tiempo (por mes, trimestre, año)?

Tendencia de ventas a lo largo del tiempo, por mes, trimestre y año.



Código visualización

```
plt.figure(figsize=(12, 4))

plt.subplot(1, 3, 1)
sns.lineplot(x='mes', y='ventas', data=ventas_por_mes)
plt.xlabel('Mes')
plt.ylabel('Total de Ventas')
plt.title('Ventas por Mes')
plt.xticks(rotation=45)

plt.subplot(1, 3, 2)
sns.lineplot(x='trimestre', y='ventas', data=ventas_por_trimestre)
plt.xlabel('Trimestre')
plt.ylabel('Total de Ventas')
plt.title('Ventas por Trimestre')
plt.xticks(rotation=45)
```

```
plt.subplot(1, 3, 3)
sns.lineplot(x='año', y='ventas', data=ventas_por_anio)
plt.xlabel('Año')
plt.ylabel('Total de Ventas')
plt.title('Ventas por Año')
plt.xticks(rotation=45)

plt.tight_layout()
plt.show()
```

Conclusiones

Cumplimiento de Objetivos: Resumir cómo el proyecto cumplió los objetivos y requisitos planteados.

El correcto desarrollo del proyecto responde a una secuencia lógica adecuada en la generación de cada una de sus partes, desde la infraestructura hasta el análisis de información.