```
# STEP 0: Setup
# Run this first to install necessary packages (if needed)
!pip install pandas matplotlib seaborn plotly scikit-learn openpyxl
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (2.2.2)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (3.10.0)
Requirement already satisfied: seaborn in /usr/local/lib/python3.11/dist-packages (0.13.2)
Requirement already satisfied: plotly in /usr/local/lib/python3.11/dist-packages (5.24.1)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.11/dist-packages (1.6.1)
Requirement already satisfied: openpyxl in /usr/local/lib/python3.11/dist-packages (3.1.5)
Requirement already satisfied: numpy>=1.23.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.0.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.2)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.3.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (4.59.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.4.8)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (25.0)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (11.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (3.2.3)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.11/dist-packages (from plotly) (8.5.0)
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (1.16.0)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (1.5.1)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (3.6.0)
Requirement already satisfied: et-xmlfile in /usr/local/lib/python3.11/dist-packages (from openpyxl) (2.0.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas) (1.17.0)
```

```
# STEP 1: Import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.cluster import KMeans
from sklearn.metrics import mean_squared_error, classification_report
from sklearn.preprocessing import LabelEncoder
import warnings
warnings.filterwarnings("ignore")
```

```
# STEP 2: Upload and load data
from google.colab import files
uploaded = files.upload()  # Upload your CSV file

df_raw = pd.read_csv(next(iter(uploaded)))
df_raw.head(10)
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving Merged_Orders_Report.csv to Merged_Orders_Report (1).csv
```

	Invoice No.	Unnamed: 1	Biller	KOT No.	Payment Type	Payment Description	Order Type	Status	Area	Sub Order Type	...	CGST	Amount (SGST).1	SGST.
0	Total	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	98841.27	4563.55	114.0
1	Min.	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	0.0	0.0	0.(
2	Max.	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	94.05	165.72	4.1
3	Avg.	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	4.19	0.19	0.(
4	163268	NaN	biller	231	Online	NaN	Delivery(Parcel)	Success	Swiggy_Thali King	Thali King - Swiggy	...	4.91	0.0	0.(
5	163267	NaN	biller	230	Online	NaN	Delivery(Parcel)	Success	Zomato	Thali King - Zomato	...	6.13	0.0	0.(
6	163266	NaN	biller	229	Online	NaN	Delivery(Parcel)	Success	Swiggy_Thali	Thali	...	4.91	0.0	0.(

```
# STEP 3: Data Cleaning (Customize this based on your file's structure)

# Drop initial non-data rows (usually metadata/header)
df = df_raw.copy()

# Find first meaningful row (likely containing actual column names)
header_row_index = df[df.iloc[:,0].str.contains('Invoice No.', na=False)].index[0]

# Set proper headers and remove non-data rows
```

```
df.columns = df.iloc[header_row_index]
df = df[(header_row_index + 1):].reset_index(drop=True)

# Drop completely empty columns
df = df.dropna(axis=1, how='all')

# Drop completely empty rows
df = df.dropna(axis=0, how='all')

# Show cleaned data
df.head(20)
```

23572	Invoice No.	Biller	KOT No.	Payment Type	Payment Description	Order Type	Status	Area	Sub Order Type	Group Name	...	CGST	Amount (SGST)
0	Total	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	117095.25	9170.89
1	Min.	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	0	0
2	Max.	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	130.05	324.3
3	Avg.	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	4.57	0.36
4	189331	Autoaccept	323	Online	NaN	Delivery(Parcel)	Success	Zomato_Wow Chinese	Wow Chinese - Zomato	Wow Chinese	...	2.5	0
5	189330	biller	322	Online	NaN	Delivery(Parcel)	Success	Zomato	Thali King - Zomato	Thali King	...	5.42	0
6	189329	biller	321	Online	NaN	Delivery(Parcel)	Success	Swiggy_Thali King	Thali King - Swiggy	Thali King	...	4.5	0
7	189328	biller	320	Online	NaN	Delivery(Parcel)	Success	Swiggy_Taste Of Thali	Taste Of Thali - Swiggy	Taste Of Thali	...	3.68	0
8	189327	biller	319	Online	NaN	Delivery(Parcel)	Success	Zomato	Thali King - Zomato	Thali King	...	4.45	0
9	189326	biller	318	Online	NaN	Delivery(Parcel)	Success	Swiggy_Thali King	Thali King - Swiggy	Thali King	...	5.13	0

```
# STEP 4: Preprocessing
# Convert 'Date' column to datetime (use actual column name for date)
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

# Convert relevant columns to numeric
numeric_cols = ['Qty', 'Rate', 'Amount', 'Discount', 'Taxable Amount']  # update based on your dataset
for col in numeric_cols:
    if col in df.columns:
        df[col] = pd.to_numeric(df[col], errors='coerce')

# Handle missing values
df = df.dropna(subset=['Date'])  # Drop rows with missing date

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 25641 entries, 4 to 25644
Data columns (total 48 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Invoice No.            25641 non-null  object
 1   Biller                 25641 non-null  object
 2   KOT No.                25640 non-null  object
 3   Payment Type           25641 non-null  object
 4   Payment Description    0 non-null      object
 5   Order Type             25641 non-null  object
 6   Status                 25641 non-null  object
 7   Area                   25640 non-null  object
 8   Sub Order Type         25641 non-null  object
 9   Group Name             25641 non-null  object
 10  Brand Name             0 non-null      object
 11  GSTIN                  212 non-null    object
```

```
 12  Assign To               10 non-null     object
 13  Phone                   1297 non-null   object
 14  Name                    21629 non-null  object
 15  Address                 10498 non-null  object
 16  Locality                67 non-null     object
 17  Persons                 0 non-null      object
 18  Order Cancel Reason     0 non-null      object
 19  My Amount (₹)           25641 non-null  object
 20  Discount (₹)            25641 non-null  object
 21  Net Sales (₹)(M.A - D)  25641 non-null  object
 22  Delivery Charge         25641 non-null  object
 23  Container Charge        25641 non-null  object
 24  Service Charge          25641 non-null  object
 25  Additional Charge       25641 non-null  object
 26  Total Tax (₹)           25641 non-null  object
 27  Round Off               25641 non-null  object
 28  Waived off              25641 non-null  object
 29  Total (₹)               25641 non-null  object
 30  Online Tax Calculated   25641 non-null  object
 31  GST Paid by Merchant    25641 non-null  object
 32  GST Paid by Ecommerce   25641 non-null  object
 33  Tip (₹)                 25641 non-null  object
 34  Non Taxable             25641 non-null  object
 35  Amount (SGST)           25641 non-null  object
 36  SGST                    25641 non-null  object
 37  Amount (CGST)           25641 non-null  object
 38  CGST                    25641 non-null  object
 39  Amount (SGST)           25641 non-null  object
 40  SGST                    25641 non-null  object
 41  Amount (CGST)           25641 non-null  object
 42  CGST                    25641 non-null  object
 43  Amount (Unknown Tax)    25641 non-null  object
 44  Unknown Tax             25641 non-null  object
 45  Date                    25641 non-null  datetime64[ns]
 46  nan                     0 non-null      object
 47  nan                     0 non-null      object
dtypes: datetime64[ns](1), object(47)
memory usage: 9.6+ MB
```

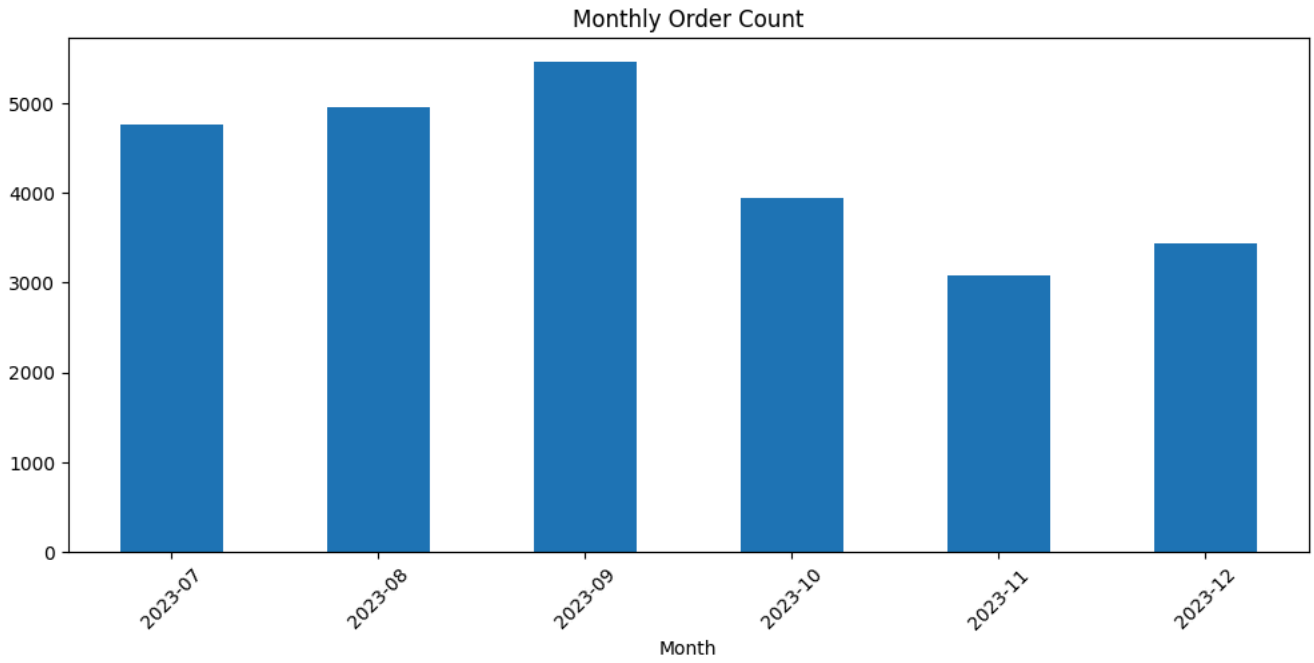
```
# Get descriptive statistics for all numeric columns
df.describe(include='all').transpose()
```

	count	unique	top	freq
23572				
Invoice No.	83396	82288	Avg.	3
Biller	83380	3	Autoaccept	65669
KOT No.	83379	598	121	313
Payment Type	83380	5	Online	69137
Payment Description	2	1	Payment Description	2
Order Type	83380	3	Delivery(Parcel)	69199
Status	83380	2	Success	83378
Area	83379	14	Zomato	28193
Sub Order Type	83380	14	Thali King - Zomato	28193
Group Name	83380	8	Thali King	61711
Brand Name	2	1	Brand Name	2
GSTIN	909	2	27ABLFS4593A1ZZ	907
Assign To	12	4	Suraj	7
Phone	3058	1180	9372165559	534
Name	71751	21238	SWIGGY	20071
Address	38673	442	Nagpur India	3240
Locality	83	13	All nagpur	27
Persons	2	1	Persons	2
Order Cancel Reason	2	1	Order Cancel Reason	2
My Amount (₹)	83392.0	2704.0	199.0	3487.0
Discount (₹)	83392	4893	0	13140
Net Sales (₹)(M.A - D)	83392.0	8169.0	149.0	1828.0
Delivery Charge	83392	80	0	41804
Container Charge	83392	135	10	16398
Service Charge	83392	3	0	41953
Additional Charge	83392	3	0	41953
Total Tax (₹)	83392	3376	0	8144
Round Off	83392	205	0	9041
Waived off	83392	7	0	41948
Total (₹)	83392	2316	166	1713
Online Tax Calculated	83392	5151	0	8143
GST Paid by Merchant	83392	29	0	41933
GST Paid by Ecommerce	83392	5152	0	8148
Tip (₹)	83392	3	0	41953
Non Taxable	83392.0	1251.0	0.0	35402.0
Amount (SGST)	83392	7853	0	8189
SGST	83392	3357	0	8189
Amount (CGST)	83392	7853	0	8189
CGST	83392	3357	0	8189
Amount (SGST)	83392	441	0	41553
SGST	83392	193	0	41553
Amount (CGST)	83392	441	0	41553
CGST	83392	193	0	41553
Amount (Unknown Tax)	83392	3	0	41953
Unknown Tax	83392	3	0	41953
Date	25641	22653	03-08-2023 13:55	12
NaN	23090	20605	06-03-2024 13:36	7
NaN	34653	29748	29-08-2024 22:20	12

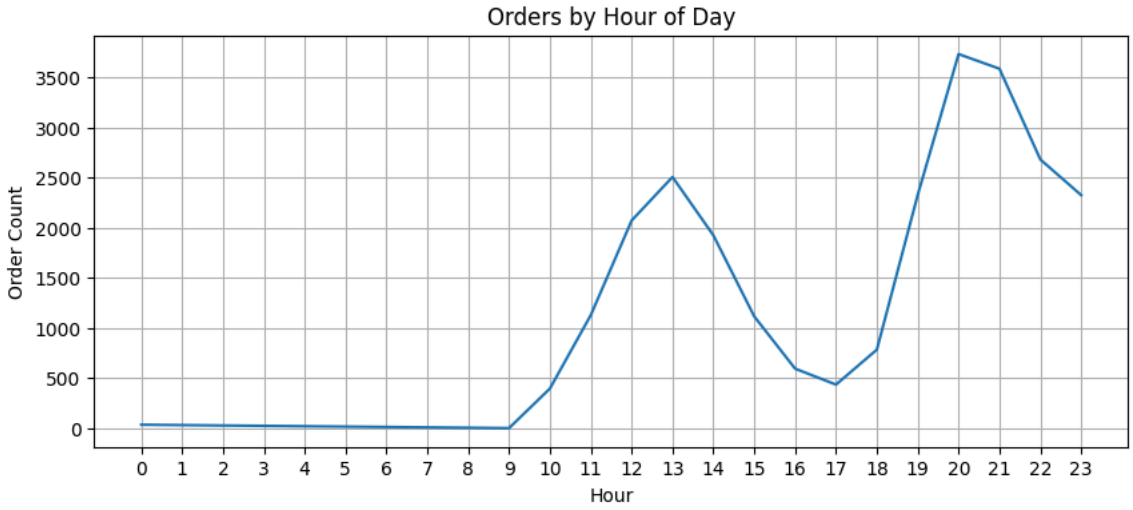
```
df.describe(include=['object'])
```

23572	Invoice No.	Biller	KOT No.	Payment Type	Payment Description	Order Type	Status	Area	Sub Order Type	Group Name	...	CGST	Amount (SGST)	SGST	Am (C
count	83396	83380	83379	83380	2	83380	83380	83379	83380	83380	...	83392	83392	83392	8
unique	82288	3	598	5	1	3	2	14	14	8	...	3357	441	193	
top	Avg.	Autoaccept	121	Online	Payment Description	Delivery(Parcel)	Success	Zomato	Thali King -	Thali King	...	0	0	0	

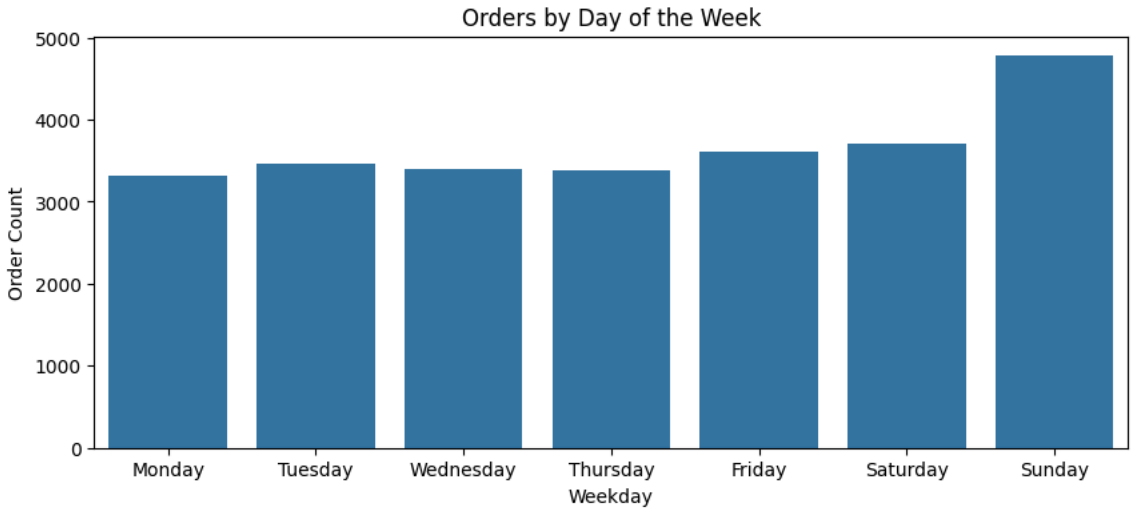
```
# STEP 5: Exploratory Data Analysis (EDA)

# Orders over time
df['Month'] = df['Date'].dt.to_period('M')
monthly_orders = df.groupby('Month').size()

monthly_orders.plot(kind='bar', figsize=(12,5), title='Monthly Order Count')
plt.xticks(rotation=45)
plt.show()
```



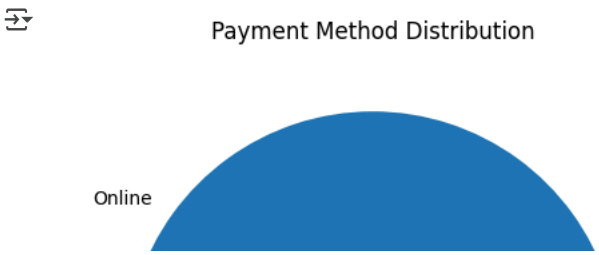

```
#Orders by Hour of Day
df['Hour'] = pd.to_datetime(df['Date'], errors='coerce').dt.hour
hourly_orders = df['Hour'].value_counts().sort_index()

plt.figure(figsize=(10,4))
sns.lineplot(x=hourly_orders.index, y=hourly_orders.values)
plt.title('Orders by Hour of Day')
plt.xlabel('Hour')
plt.ylabel('Order Count')
plt.xticks(range(24))
plt.grid(True)
plt.show()
```



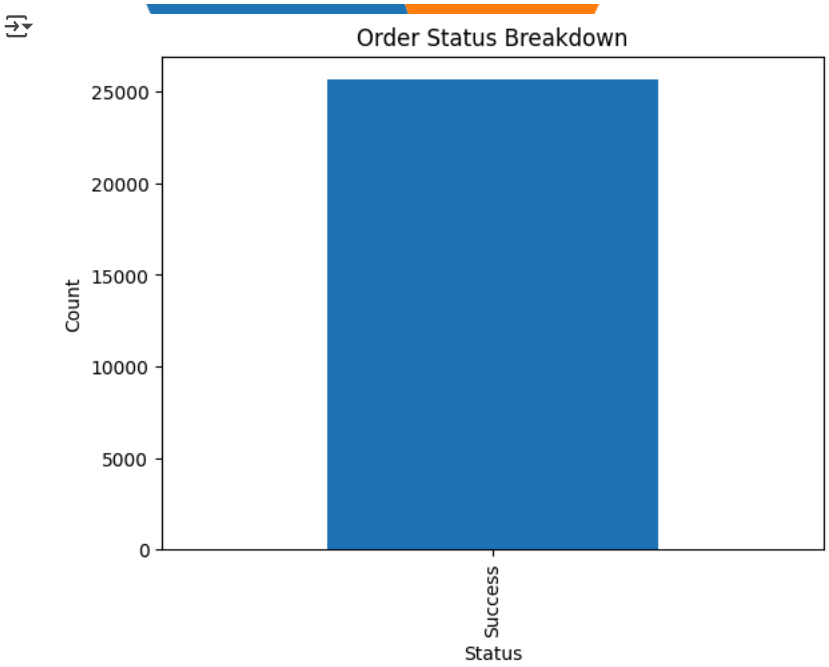

```
#Orders by Day of Week
df['Weekday'] = pd.to_datetime(df['Date'], errors='coerce').dt.day_name()
weekday_orders = df['Weekday'].value_counts().reindex([
    'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday'])

plt.figure(figsize=(10,4))
sns.barplot(x=weekday_orders.index, y=weekday_orders.values)
plt.title('Orders by Day of the Week')
plt.ylabel('Order Count')
plt.show()
```




```
#Payment Method Distribution
if 'Payment Type' in df.columns:
    df['Payment Type'].value_counts().plot(kind='pie', autopct='%1.1f%%', figsize=(6,6))
    plt.title('Payment Method Distribution')
    plt.ylabel('')
    plt.show()
```




```
# Order Status Analysis (Cancelled, Success, etc.)
if 'Status' in df.columns:
    status_counts = df['Status'].value_counts()
    status_counts.plot(kind='bar', title='Order Status Breakdown')
    plt.ylabel('Count')
    plt.show()
```




```
#Heatmap: Orders by Day and Hour
df['Day'] = pd.to_datetime(df['Date'], errors='coerce').dt.day_name()
df['Hour'] = pd.to_datetime(df['Date'], errors='coerce').dt.hour

heatmap_data = df.groupby(['Day', 'Hour']).size().unstack().reindex(
    ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday'])

plt.figure(figsize=(14,6))
sns.heatmap(heatmap_data, cmap="YlGnBu")
```