Data Import

In [4]:
```
import pandas as pd
import numpy as np
df = pd.read_csv("C:/Users/JOSH/Desktop/60 Days Challenge/Project 4/glol
                 parse_dates=['Transaction Date'])
```

Data Info and Description

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500000 entries, 0 to 499999
Data columns (total 10 columns):
 #   Column            Non-Null Count   Dtype         
---  ------            --------------   -----         
 0   Transaction Date  500000 non-null  datetime64[ns]
 1   Customer ID       500000 non-null  object        
 2   Region            500000 non-null  object        
 3   Product ID        500000 non-null  object        
 4   Category          500000 non-null  object        
 5   Price             500000 non-null  float64       
 6   Quantity          500000 non-null  int64         
 7   Discount (%)      500000 non-null  float64       
 8   Total Revenue     500000 non-null  float64       
 9   Payment Method    500000 non-null  object        
dtypes: datetime64[ns](1), float64(3), int64(1), object(5)
memory usage: 38.1+ MB
```

In [6]:
```
df.describe()
```

Out[6]:

	Price	Quantity	Discount (%)	Total Revenue
count	500000.000000	500000.000000	500000.000000	500000.000000
mean	502.732868	3.001728	14.973566	1282.473695
std	286.922294	1.414259	8.667876	1022.572960
min	5.000000	1.000000	0.000000	3.580000
25%	254.980000	2.000000	7.450000	469.740000
50%	502.300000	3.000000	14.980000	993.315000
75%	751.260000	4.000000	22.470000	1890.542500
max	999.990000	5.000000	30.000000	4988.910000

Checking for Missing Values

In [7]:
```
print("Missing values count:")
print(df.isnull().sum())
```

```
Missing values count:
Transaction Date    0
Customer ID         0
Region              0
Product ID          0
Category            0
Price               0
Quantity            0
Discount (%)        0
Total Revenue       0
Payment Method      0
dtype: int64
```

Confrming Column Data Types

In [8]:
```
print("\nColumn data types:")
print(df.dtypes)
```

```
Column data types:
Transaction Date    datetime64[ns]
Customer ID                 object
Region                      object
Product ID                  object
Category                    object
Price                      float64
Quantity                     int64
Discount (%)               float64
Total Revenue              float64
Payment Method              object
dtype: object
```

Detecting Outliers

In [20]:
```
# Function to detect and flag outliers
def flag_outliers(series):
    Q1 = series.quantile(0.25)
    Q3 = series.quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    outlier_flags = np.where((series < lower_bound) | (series > upper_b
    return outlier_flags
```

In [22]:

```
# Flag otutliers in quantiy, price, tota_revenue
for col in ['Quantity', 'Price', 'Total Revenue']:
    df[f'{col}_outlier'] = flag_outliers(df[col])

print("\nOutliers:")
print("Outliers in Quantity:")
print(df[df['Quantity_outlier'] == 1])
print("\nOutliers in Price:")
print(df[df['Price_outlier'] == 1])
print("\nOutliers in Total Revenue:")
print(df[df['Total Revenue_outlier'] == 1])
```

```
Outliers:
Outliers in Quantity:
Empty DataFrame
Columns: [Transaction Date, Customer ID, Region, Product ID, Category, Pri
ce, Quantity, Discount (%), Total Revenue, Payment Method, Quantity_outlie
r, Price_outlier, Total Revenue_outlier]
Index: []

Outliers in Price:
Empty DataFrame
Columns: [Transaction Date, Customer ID, Region, Product ID, Category, Pri
ce, Quantity, Discount (%), Total Revenue, Payment Method, Quantity_outlie
r, Price_outlier, Total Revenue_outlier]
Index: []

Outliers in Total Revenue:
       Transaction Date Customer ID         Region    Product ID  \
19           2022-11-09  CUST_13923      Australia  Product_4949
149          2022-04-15  CUST_54133  South America  Product_7213
275          2023-09-26  CUST_91585         Africa  Product_2701
289          2023-02-18  CUST_61249  South America  Product_7914
334          2023-10-21  CUST_32126      Australia  Product_2237
...                 ...         ...            ...          ...
499459       2022-10-21  CUST_60201         Africa  Product_1720
499539       2022-12-22  CUST_25686  North America  Product_6801
499733       2023-07-13  CUST_13131      Australia  Product_3921
499766       2023-07-23  CUST_53197  South America  Product_3607
499886       2023-01-20  CUST_36833         Africa  Product_1887

              Category   Price  Quantity  Discount (%)  Total Revenue  \
19             Fashion  981.20         5          0.80        4866.75
149     Health & Beauty  845.97         5          4.82        4025.97
275               Books  961.43         5          6.66        4486.99
289        Toys & Games  991.94         5         12.94        4317.91
334             Fashion  981.44         5          5.15        4654.48
...                 ...     ...       ...           ...           ...
499459     Toys & Games  927.11         5          8.48        4242.46
499539          Fashion  989.82         5         13.52        4279.98
499733     Toys & Games  956.76         5         13.16        4154.25
499766      Electronics  839.83         5          0.36        4184.03
499886   Home & Kitchen  917.00         5          1.23        4528.60

       Payment Method  Quantity_outlier  Price_outlier  Total Revenue_outl
ier
19      Bank Transfer                 0              0
1
149       Credit Card                 0              0
1
275              Cash                 0              0
1
289     Bank Transfer                 0              0
1
334            Crypto                 0              0
1
...               ...               ...            ...
...
499459         PayPal                 0              0
1
499539    Credit Card                 0              0
1
```

```
499733         Crypto                 0              0
1
499766    Credit Card                 0              0
1
499886  Bank Transfer                 0              0
1

[6877 rows x 13 columns]
```

Creating Time_Based Features

In [24]:
```
df['year'] = df['Transaction Date'].dt.year
df['month'] = df['Transaction Date'].dt.month
df['day'] = df['Transaction Date'].dt.day
df['day_of_week'] = df['Transaction Date'].dt.day_name()
```

Basic Exploratory Stats

Unique Count of Customers

In [26]:
```
df['Customer ID'].nunique()
```

Out[26]: 98348

Unique Count of Products

In [27]:
```
df['Product ID'].nunique()
```

Out[27]: 10000

Descriptive Stats

In [29]:
```
df[['Price', 'Quantity', 'Discount (%)', 'Total Revenue']].describe()
```

Out[29]:

	Price	Quantity	Discount (%)	Total Revenue
count	500000.000000	500000.000000	500000.000000	500000.000000
mean	502.732868	3.001728	14.973566	1282.473695
std	286.922294	1.414259	8.667876	1022.572960
min	5.000000	1.000000	0.000000	3.580000
25%	254.980000	2.000000	7.450000	469.740000
50%	502.300000	3.000000	14.980000	993.315000
75%	751.260000	4.000000	22.470000	1890.542500
max	999.990000	5.000000	30.000000	4988.910000