Data Cleaning with Python

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
import warnings
warnings.filterwarnings('ignore')
import re
```

Data Collection and Inspection

In [2]:
```
# file reading
df = pd.read_csv('messy_customer_sales_data.csv')
df
```

Out[2]:

	Customer_ID	Name	Gender	Age	City	Signup_Date	Last_Pur
0	CUST4371	Paul Wilson	m	52.0	KOLKATA	2025-06-26	
1	CUST5957	Jason Thomas	M	51.0 years	NaN	2021-02-17	
2	CUST3754	Brittney Martinez	F	62.0	hyderabad	2023-11-05	
3	CUST2934	Brenda Pierce	FEMALE	40.0	hyderabad	2022-03-13	
4	CUST5683	Matthew Carroll	f	41.0	CHENNAI	2024-04-05	
...	...	...	...	...	...	...	
10195	CUST10767	Robert Lewis	female	35.0 years	delhi	2020-12-08	
10196	NaN	Diane Evans	M	53.0	bangalore	2023-12-31	
10197	CUST6315	Joshua Martinez	m	25.0	hyderabad	2022-02-15	
10198	CUST4812	Sarah Miller	FEMALE	55.0	NaN	2021-03-16	
10199	CUST6588	David Potter	female	34.0	HYDERABAD	2020-10-12	

10200 rows × 12 columns

In [3]:
```
# data types
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10200 entries, 0 to 10199
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   Customer_ID         9177 non-null   object 
 1   Name                10200 non-null  object 
 2   Gender              9174 non-null   object 
 3   Age                 9249 non-null   object 
 4   City                9184 non-null   object 
 5   Signup_Date         10200 non-null  object 
 6   Last_Purchase_Date  9188 non-null   object 
 7   Purchase_Amount     9179 non-null   float64
 8   Feedback_Score      9177 non-null   float64
 9   Email               10200 non-null  object 
 10  Phone_Number        10200 non-null  int64  
 11  Country             9468 non-null   object 
dtypes: float64(2), int64(1), object(9)
memory usage: 956.4+ KB
```

In [4]:
```
# summary statistics
df.describe()
```

Out[4]:

	Purchase_Amount	Feedback_Score	Phone_Number
count	9.179000e+03	9177.000000	1.020000e+04
mean	2.909013e+04	5.479351	4.979974e+09
std	2.086971e+05	2.867123	2.902593e+09
min	-5.000000e+02	1.000000	9.208990e+05
25%	1.229500e+04	3.000000	2.449157e+09
50%	2.433000e+04	5.000000	4.988639e+09
75%	3.713000e+04	8.000000	7.510448e+09
max	9.999999e+06	10.000000	9.994402e+09

In [5]:
```
# missing value summary
df.isnull().sum()
```

```
Out[5]: Customer_ID           1023
        Name                     0
        Gender                1026
        Age                    951
        City                  1016
        Signup_Date              0
        Last_Purchase_Date    1012
        Purchase_Amount       1021
        Feedback_Score        1023
        Email                    0
        Phone_Number             0
        Country                732
        dtype: int64
```

```
In [6]: # total number of row & columns
        df.shape
```

```
Out[6]: (10200, 12)
```

```
In [7]: df.dropna().shape
```

```
Out[7]: (4528, 12)
```

```
In [8]: # duplicate rows summary
        df[df.duplicated()].shape
```

```
Out[8]: (15, 12)
```

```
In [9]: df[df.duplicated()]
```

Out[9]:

	Customer_ID	Name	Gender	Age	City	Signup_Date	Last_Purc
2032	CUST2755	Casey Campbell	male	NaN	Hyderabad	2023-12-01	
3191	NaN	Travis Schneider	f	51.0	NaN	2025-04-20	
4312	CUST8305	Stanley Cain	FEMALE	51.0	Bangalore	2023-08-14	
5611	CUST6288	Jonathon Kim	male	46.0	Chennai	2023-11-22	
5680	CUST2695	Lisa Durham	NaN	46.0	Chennai	2021-04-18	
5899	CUST8841	Amanda Hill	NaN	34.0	chennai	2022-09-11	
6070	CUST10780	Emily Smith	NaN	23.0	Kolkata	2023-12-09	
6240	CUST9745	David Morales	NaN	56.0	NaN	2023-02-18	
6425	CUST4631	George Villa	m	26.0	Mumbai	2022-05-21	
6498	CUST1711	Stephanie Elliott	NaN	44.0	MUMBAI	2023-05-17	
7118	CUST9380	Paul Wilson	MALE	42.0	Bangalore	2025-07-31	
8392	CUST1436	Ralph Anderson	female	33.0	Kolkata	2022-10-21	
9155	CUST10040	Mark Taylor	NaN	43.0	Delhi	2021-04-16	
9960	CUST6220	John Rodriguez	male	51.0	Hyderabad	2021-03-28	
10140	CUST4367	Amber Kennedy	F	61.0	Bangalore	2023-11-05	

In [10]:
```
df['Customer_ID'].value_counts()
```

Out[10]:
```
Customer_ID
CUST3344     2
CUST4893     2
CUST7000     2
CUST10824    2
CUST2695     2
            ..
CUST7857     1
CUST3881     1
CUST1565     1
CUST9038     1
CUST5957     1
Name: count, Length: 9000, dtype: int64
```

In [11]:
```python
df[df['Customer_ID'] == 'CUST7000']
```

Out[11]:

	Customer_ID	Name	Gender	Age	City	Signup_Date	Last_Purchase_Da
1508	CUST7000	Brittany Ortiz	FEMALE	NaN	Delhi	2022-03-27	2025-02-
3194	CUST7000	Brittany Ortiz	f	NaN	Delhi	2022-03-27	2025-02-

In [12]:
```python
# category count
for col in df.columns:
    if df[col].nunique() < 20:
        print(df[col].value_counts())
        print('-'*50)
```

```
Gender
f         1184
M         1171
m         1163
F         1157
MALE      1131
female    1128
male      1121
FEMALE    1119
Name: count, dtype: int64
--------------------------------------------------
City
Kolkata       820
Mumbai        812
Chennai       784
Bangalore     773
Hyderabad     770
Delhi         763
CHENNAI       404
KOLKATA       395
MUMBAI        393
hyderabad     384
bangalore     383
DELHI         378
delhi         369
BANGALORE     363
HYDERABAD     360
mumbai        352
chennai       343
kolkata       338
Name: count, dtype: int64
--------------------------------------------------
Feedback_Score
2.0     952
4.0     947
7.0     938
6.0     927
3.0     913
8.0     912
1.0     907
9.0     903
10.0    901
5.0     877
Name: count, dtype: int64
--------------------------------------------------
Country
India    7132
IND       793
india     772
InDia     771
Name: count, dtype: int64
--------------------------------------------------
```

Handling missing data

```
In [13]: # Drop rows with missing 'Customer_ID' (unique identifier)
         df.dropna(subset = ['Customer_ID'], inplace = True)
```

```
In [14]: df.isnull().sum()
```

```
Out[14]: Customer_ID              0
         Name                     0
         Gender                 934
         Age                    859
         City                   918
         Signup_Date              0
         Last_Purchase_Date     914
         Purchase_Amount        927
         Feedback_Score         905
         Email                    0
         Phone_Number             0
         Country                664
         dtype: int64
```

```
In [15]: df['Age'].unique()
```

```
Out[15]: array(['52.0', '51.0 years', '62.0', '40.0', '41.0', nan, '18.0',
                '43.0 years', '40.0 years', '26.0', '32.0', '22.0', '59.0', '65.0',
                '61.0', '31.0', '54.0 years', '55.0', '69.0', '61.0 years', '24.0',
                '63.0', '19.0', '50.0', '56.0', '36.0', '68.0', '43.0', '38.0',
                '27.0', '57.0 years', '23.0', '25.0', '66.0', '28.0', '30.0',
                '46.0', '48.0', '20.0', '37.0', '67.0', '51.0', '35.0', '58.0',
                '29.0', 'nan years', '39.0', '49.0', '47.0', '42.0', '44.0',
                '64.0', '53.0', '60.0', '59.0 years', '45.0', '21.0', '34.0',
                '54.0', '48.0 years', '46.0 years', '33.0', '57.0', '30.0 years',
                '58.0 years', '35.0 years', '34.0 years', '69.0 years', '250',
                '19.0 years', '27.0 years', '53.0 years', '65.0 years',
                '66.0 years', '44.0 years', '49.0 years', '25.0 years',
                '23.0 years', '62.0 years', '41.0 years', '33.0 years',
                '28.0 years', '22.0 years', '20.0 years', '42.0 years',
                '45.0 years', '3', '63.0 years', '37.0 years', '38.0 years',
                '55.0 years', '18.0 years', '36.0 years', '67.0 years',
                '29.0 years', '39.0 years', '31.0 years', '47.0 years',
                '52.0 years', '-10', '60.0 years', '24.0 years', '26.0 years',
                '21.0 years', '64.0 years', '50.0 years', '68.0 years',
                '32.0 years', '56.0 years'], dtype=object)
```

```
In [16]: def extract_age(age):
             age_num = re.findall('[0-9]+', str(age))
             if len(age_num) > 0:
                 return age_num[0]
             else:
                 return age
```

```
df['Age'] = df['Age'].apply(lambda x: extract_age(x))
```

```
In [17]: df_age = df[df['Age'] != 'nan years']['Age']
```

```
In [18]: age_median = int(df_age.dropna().astype('int64').median())
```

```
In [19]: age_median

Out[19]: 43
```

```
In [20]: # replace nan with median age and extracting numbers
         df['Age'].replace('nan years', age_median, inplace = True)
```

```
In [21]: df['Age'].fillna(age_median, inplace = True)
```

```
In [22]: df['Age'].unique()

Out[22]: array(['52', '51', '62', '40', '41', 43, '18', '43', '26', '32', '22',
                '59', '65', '61', '31', '54', '55', '69', '24', '63', '19', '50',
                '56', '36', '68', '38', '27', '57', '23', '25', '66', '28', '30',
                '46', '48', '20', '37', '67', '35', '58', '29', '39', '49', '47',
                '42', '44', '64', '53', '60', '45', '21', '34', '33', '250', '3',
                '10'], dtype=object)
```

```
In [23]: df['Purchase_Amount'].fillna(df['Purchase_Amount'].median(), inplace = True)
```

```
In [24]: df['Feedback_Score'].fillna(df['Feedback_Score'].mode()[0], inplace = True)
```

```
In [25]: df.isnull().sum()

Out[25]: Customer_ID             0
         Name                    0
         Gender                934
         Age                     0
         City                  918
         Signup_Date             0
         Last_Purchase_Date    914
         Purchase_Amount         0
         Feedback_Score          0
         Email                   0
         Phone_Number            0
         Country               664
         dtype: int64
```

```
In [26]: for col in ['Gender', 'City', 'Country']:
             df[col].fillna(df[col].mode()[0], inplace = True)
```

```
In [27]: df['Last_Purchase_Date'].ffill(inplace = True)
```

Fixing Inconsistent Formatting

```
In [28]: df['Gender'].unique()
```

```
Out[28]: array(['m ', 'M', 'F', 'FEMALE', 'f ', 'male', 'MALE', 'female'],
               dtype=object)
```

```
In [29]: df['Gender'] = df['Gender'].str.strip().str.lower()
```

```
In [30]: df['Gender'].replace({'m':'male','f':'female'}, inplace = True)
```

```
In [31]: df['City'] = df['City'].str.lower().str.strip()
```

```
In [32]: df['Country'] = df['Country'].str.lower()
```

```
In [33]: df['Country'].replace({'ind':'india'}, inplace = True)
```

```
In [34]: df
```

Out[34]:

	Customer_ID	Name	Gender	Age	City	Signup_Date	Last_Purc
0	CUST4371	Paul Wilson	male	52	kolkata	2025-06-26	
1	CUST5957	Jason Thomas	male	51	kolkata	2021-02-17	
2	CUST3754	Brittney Martinez	female	62	hyderabad	2023-11-05	
3	CUST2934	Brenda Pierce	female	40	hyderabad	2022-03-13	
4	CUST5683	Matthew Carroll	female	41	chennai	2024-04-05	
...	...	...	...	...	...	...	
10194	CUST6146	Cody Thompson	female	40	kolkata	2024-08-21	
10195	CUST10767	Robert Lewis	female	35	delhi	2020-12-08	
10197	CUST6315	Joshua Martinez	male	25	hyderabad	2022-02-15	
10198	CUST4812	Sarah Miller	female	55	kolkata	2021-03-16	
10199	CUST6588	David Potter	female	34	hyderabad	2020-10-12	

9177 rows × 12 columns

Handling Duplicates

In [35]:
```
df[df.duplicated()].shape
```

Out[35]:
```
(163, 12)
```

In [36]:
```
df.drop_duplicates(inplace = True)
```

In [37]:
```
df
```

Out[37]:

	Customer_ID	Name	Gender	Age	City	Signup_Date	Last_Purc
0	CUST4371	Paul Wilson	male	52	kolkata	2025-06-26	
1	CUST5957	Jason Thomas	male	51	kolkata	2021-02-17	
2	CUST3754	Brittney Martinez	female	62	hyderabad	2023-11-05	
3	CUST2934	Brenda Pierce	female	40	hyderabad	2022-03-13	
4	CUST5683	Matthew Carroll	female	41	chennai	2024-04-05	
...	...	...	...	...	...	...	
10193	CUST6352	Isaiah Terry	female	26	kolkata	2023-03-04	
10194	CUST6146	Cody Thompson	female	40	kolkata	2024-08-21	
10195	CUST10767	Robert Lewis	female	35	delhi	2020-12-08	
10197	CUST6315	Joshua Martinez	male	25	hyderabad	2022-02-15	
10198	CUST4812	Sarah Miller	female	55	kolkata	2021-03-16	

9014 rows × 12 columns

In [38]:
```
df['Customer_ID'].value_counts()
```

Out[38]:
```
Customer_ID
CUST3693    2
CUST6833    2
CUST5341    2
CUST1002    2
CUST9950    2
           ..
CUST7857    1
CUST3881    1
CUST1565    1
CUST9038    1
CUST5957    1
Name: count, Length: 9000, dtype: int64
```

In [39]:
```python
df[df['Customer_ID'] == 'CUST3693']
```

Out[39]:

	Customer_ID	Name	Gender	Age	City	Signup_Date	Last_Purcha
3967	CUST3693	Chad Dominguez	female	3	chennai	2022-01-20	20
4953	CUST3693	Chad Dominguez	female	54	chennai	2022-01-20	20

In [40]:
```python
df.drop_duplicates(subset = ['Customer_ID'], keep = 'first')
```

Out[40]:

	Customer_ID	Name	Gender	Age	City	Signup_Date	Last_Purc
0	CUST4371	Paul Wilson	male	52	kolkata	2025-06-26	
1	CUST5957	Jason Thomas	male	51	kolkata	2021-02-17	
2	CUST3754	Brittney Martinez	female	62	hyderabad	2023-11-05	
3	CUST2934	Brenda Pierce	female	40	hyderabad	2022-03-13	
4	CUST5683	Matthew Carroll	female	41	chennai	2024-04-05	
...	...	...	...	...	...	...	
10193	CUST6352	Isaiah Terry	female	26	kolkata	2023-03-04	
10194	CUST6146	Cody Thompson	female	40	kolkata	2024-08-21	
10195	CUST10767	Robert Lewis	female	35	delhi	2020-12-08	
10197	CUST6315	Joshua Martinez	male	25	hyderabad	2022-02-15	
10198	CUST4812	Sarah Miller	female	55	kolkata	2021-03-16	

9000 rows × 12 columns

Correcting Data Types

In [41]:
```
df.columns
```

Out[41]:
```
Index(['Customer_ID', 'Name', 'Gender', 'Age', 'City', 'Signup_Date',
       'Last_Purchase_Date', 'Purchase_Amount', 'Feedback_Score', 'Email',
       'Phone_Number', 'Country'],
      dtype='object')
```

In [42]:
```
df['Age'] = df['Age'].astype('int64')
```

In [43]:
```
df['Signup_Date'] = pd.to_datetime(df['Signup_Date'])
```

In [44]:
```
df['Last_Purchase_Date'] = pd.to_datetime(df['Last_Purchase_Date'])
```

In [45]:
```
df.dtypes
```

```
Out[45]: Customer_ID                   object
         Name                          object
         Gender                        object
         Age                            int64
         City                          object
         Signup_Date           datetime64[ns]
         Last_Purchase_Date    datetime64[ns]
         Purchase_Amount              float64
         Feedback_Score               float64
         Email                         object
         Phone_Number                   int64
         Country                       object
         dtype: object
```

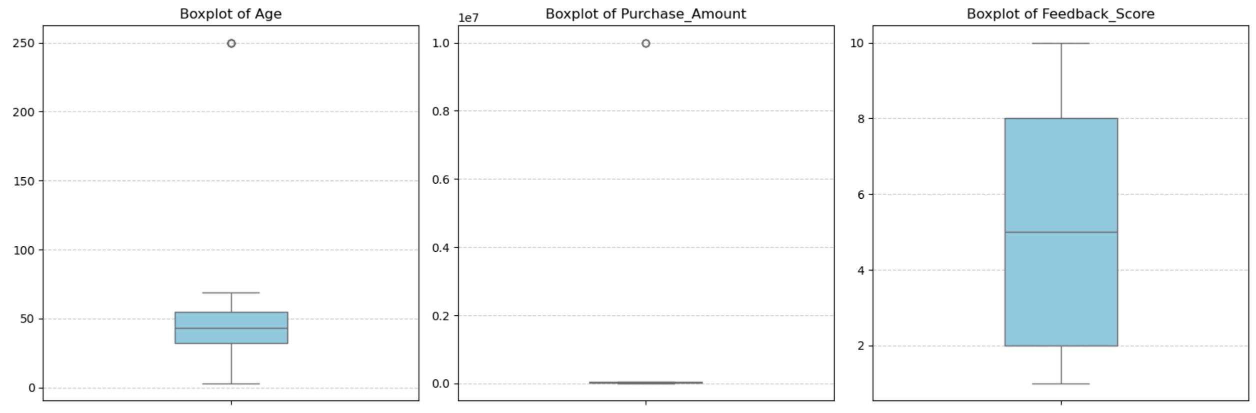
Handling Outliers

```
In [46]: cols = ['Age', 'Purchase_Amount', 'Feedback_Score']

         plt.figure(figsize = (15,5))

         for i,col in enumerate(cols,1):
             plt.subplot(1,3,i)
             sns.boxplot(y=df[col], color = 'skyblue', width = 0.3)
             plt.title(f'Boxplot of {col}', fontsize = 12)
             plt.ylabel('')
             plt.grid(axis = 'y', linestyle = '--', alpha = 0.6)

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



```
In [47]: # Calculate Z-scores
         z_scores = np.abs(stats.zscore(df[['Age', 'Purchase_Amount']]))
```

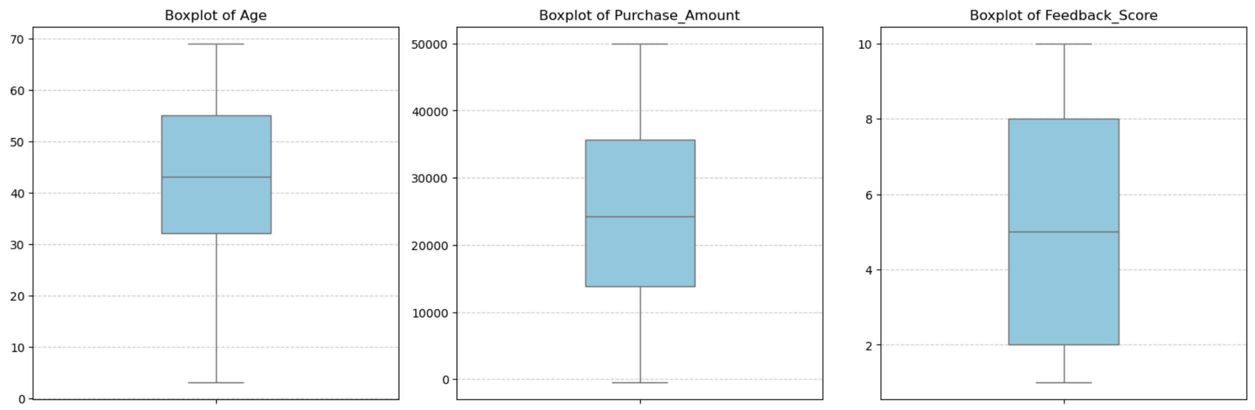
```
In [48]: # Identify outliers (any row with z > 3)
         df_clean = df[~(z_scores > 3).any(axis = 1)]
```

```
In [49]: cols = ['Age', 'Purchase_Amount', 'Feedback_Score']
```

```python
plt.figure(figsize = (15,5))

for i,col in enumerate(cols,1):
    plt.subplot(1,3,i)
    sns.boxplot(y=df_clean[col], color = 'skyblue', width = 0.3)
    plt.title(f'Boxplot of {col}', fontsize = 12)
    plt.ylabel('')
    plt.grid(axis = 'y', linestyle = '--', alpha = 0.6)

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


Boxplot of Age
Boxplot of Purchase_Amount
Boxplot of Feedback_Score