In [97]:
```
import pandas as pd
import numpy as np
```

In [98]:
```
cafe_df = pd.read_csv('C:/Users/Mr.Ishan/Downloads/Cafe Sales Dirty/dirty_cafe
_sales.csv')
```

In [4]:
```
cafe_df.shape
```

Out[4]:
```
(10000, 8)
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 8 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   Transaction ID    10000 non-null  object
 1   Item              9667 non-null   object
 2   Quantity          9862 non-null   object
 3   Price Per Unit    9821 non-null   object
 4   Total Spent       9827 non-null   object
 5   Payment Method    7421 non-null   object
 6   Location          6735 non-null   object
 7   Transaction Date  9841 non-null   object
dtypes: object(8)
memory usage: 625.1+ KB
```

In [99]:
```
cafe_df[cafe_df['Item'].isna()]
```

Out[99]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
8	TXN_4717867	NaN	5	3.0	15.0	NaN	Takeaway	2023-07-28
30	TXN_1736287	NaN	5	2.0	10.0	Digital Wallet	NaN	2023-06-02
61	TXN_8051289	NaN	1	3.0	3.0	NaN	In-store	2023-10-09
72	TXN_6044979	NaN	1	1.0	1.0	Cash	In-store	2023-12-08
89	TXN_4132730	NaN	5	1.0	5.0	NaN	In-store	2023-03-12
...	...	...	...	...	...	...	...	...
9820	TXN_8751702	NaN	5	NaN	15.0	Cash	NaN	2023-02-13
9855	TXN_3740505	NaN	2	1.5	3.0	NaN	NaN	2023-11-21
9876	TXN_3105633	NaN	1	2.0	2.0	NaN	In-store	2023-03-30
9885	TXN_4659954	NaN	3	4.0	12.0	Credit Card	In-store	NaN
9996	TXN_9659401	NaN	3	NaN	3.0	Digital Wallet	NaN	2023-06-02

333 rows × 8 columns

In [100]:
```
cafe_df['Item'].value_counts()
```
Out[100]:
```
Item
Juice       1171
Coffee      1165
Salad       1148
Cake        1139
Sandwich    1131
Smoothie    1096
Cookie      1092
Tea         1089
UNKNOWN      344
ERROR        292
Name: count, dtype: int64
```

```
In [101]: cafe_df.groupby(['Price Per Unit','Item']).agg({
              'Transaction ID':'count'
          })
```

Out[101]:

		Transaction ID
Price Per Unit	**Item**	
1.0	**Cookie**	1026
	ERROR	34
	UNKNOWN	45
1.5	**ERROR**	37
	Tea	1023
	UNKNOWN	40
2.0	**Coffee**	1108
	ERROR	31
	UNKNOWN	49
3.0	**Cake**	1085
	ERROR	77
	Juice	1110
	UNKNOWN	77
4.0	**ERROR**	61
	Sandwich	1082
	Smoothie	1036
	UNKNOWN	70
5.0	**ERROR**	39
	Salad	1082
	UNKNOWN	45
ERROR	**Cake**	19
	Coffee	18
	Cookie	21
	ERROR	3
	Juice	26
	Salad	34
	Sandwich	13
	Smoothie	19
	Tea	25
	UNKNOWN	4

		Transaction ID
Price Per Unit	**Item**	
UNKNOWN	**Cake**	14
	Coffee	20
	Cookie	21
	ERROR	3
	Juice	18
	Salad	16
	Sandwich	19
	Smoothie	17
	Tea	21
	UNKNOWN	7

In [102]:
```
mask = (cafe_df['Item'].isin(['ERROR','UNKNOWN'])) & (cafe_df['Price Per Uni
t'] == '1.0')

cafe_df.loc[mask, 'Item'] = 'Cookie'
```

In [103]:
```
mask.sum()
```

Out[103]:
```
np.int64(79)
```

In [104]:
```
cafe_df[cafe_df['Item'].isin(['ERROR','UNKNOWN'])]
```

Out[104]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
6	TXN_4433211	UNKNOWN	3	3.0	9.0	ERROR	Takeaway	2023-10-06
14	TXN_8915701	ERROR	2	1.5	3.0	NaN	In-store	2023-03-21
36	TXN_6855453	UNKNOWN	4	3.0	12.0	NaN	In-store	2023-07-17
52	TXN_8914892	UNKNOWN	5	5.0	25.0	Digital Wallet	NaN	2023-03-15
63	TXN_9099694	UNKNOWN	3	5.0	15.0	NaN	Takeaway	2023-11-18
...	...	...	...	...	...	...	...	...
9918	TXN_2292088	ERROR	1	4.0	4.0	Digital Wallet	Takeaway	2023-03-04
9946	TXN_8807600	UNKNOWN	1	4.0	4.0	Cash	Takeaway	2023-09-24
9958	TXN_4125474	ERROR	2	5.0	10.0	Credit Card	In-store	2023-08-02
9981	TXN_4583012	ERROR	5	4.0	20.0	Digital Wallet	NaN	2023-02-27
9994	TXN_7851634	UNKNOWN	4	4.0	16.0	NaN	NaN	2023-01-08

557 rows × 8 columns

```
In [105]: cafe_df.groupby(['Price Per Unit','Item']).agg({
              'Transaction ID':'count'
          })
```

Out[105]:

		Transaction ID
Price Per Unit	**Item**	
1.0	**Cookie**	1105
1.5	**ERROR**	37
	Tea	1023
	UNKNOWN	40
2.0	**Coffee**	1108
	ERROR	31
	UNKNOWN	49
3.0	**Cake**	1085
	ERROR	77
	Juice	1110
	UNKNOWN	77
4.0	**ERROR**	61
	Sandwich	1082
	Smoothie	1036
	UNKNOWN	70
5.0	**ERROR**	39
	Salad	1082
	UNKNOWN	45
ERROR	**Cake**	19
	Coffee	18
	Cookie	21
	ERROR	3
	Juice	26
	Salad	34
	Sandwich	13
	Smoothie	19
	Tea	25
	UNKNOWN	4

		Transaction ID
Price Per Unit	**Item**	
UNKNOWN	**Cake**	14
	Coffee	20
	Cookie	21
	ERROR	3
	Juice	18
	Salad	16
	Sandwich	19
	Smoothie	17
	Tea	21
	UNKNOWN	7

```
In [106]: cafe_df[cafe_df['Item'] == 'Cake']
```

Out[106]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
1	TXN_4977031	Cake	4	3.0	12.0	Cash	In-store	2023-05-16
18	TXN_8876618	Cake	5	3.0	15.0	Cash	ERROR	2023-03-25
29	TXN_7640952	Cake	4	3.0	12.0	Digital Wallet	Takeaway	ERROR
49	TXN_8230936	Cake	3	3.0	9.0	NaN	ERROR	2023-05-02
50	TXN_7742742	Cake	5	3.0	15.0	NaN	Takeaway	2023-09-05
...	...	...	...	...	...	...	...	...
9964	TXN_8938445	Cake	3	3.0	9.0	NaN	In-store	2023-11-07
9972	TXN_3124078	Cake	4	3.0	12.0	UNKNOWN	In-store	2023-08-06
9975	TXN_9668108	Cake	1	3.0	3.0	Cash	In-store	2023-01-20
9985	TXN_3297457	Cake	2	3.0	6.0	NaN	UNKNOWN	2023-01-03
9988	TXN_9594133	Cake	5	3.0	NaN	ERROR	NaN	NaN

1139 rows × 8 columns

```
In [107]: #Step1 Replace Unknown,Error in Item by matching price per unit

          #Step2 Replace Unknown, Error in Price per unit by matching Item
          #Step3 Replace nan,Unknown,Error in Payment method and Location by current val
          ues
```

```
In [108]: cafe_df.columns
```

```
Out[108]: Index(['Transaction ID', 'Item', 'Quantity', 'Price Per Unit', 'Total Spent',
                 'Payment Method', 'Location', 'Transaction Date'],
                dtype='object')
```

In [109]:
```
price_to_item = {
    1.5: "Tea",
    2: "Coffee",
    5:"Salad"

}
```

In [110]:
```
mask = cafe_df['Item'].isin([pd.NA, 'UNKNOWN', 'ERROR']) | cafe_df['Item'].isn
a()

cafe_df.loc[mask, 'Item'] = cafe_df.loc[mask].apply(
    lambda row: price_to_item.get(row['Price Per Unit'], row['Item']),
    axis=1
)
```

```
In [111]: cafe_df.groupby(['Price Per Unit','Item']).agg({
              'Transaction ID':'count'
          })
```

Out[111]:

		Transaction ID
Price Per Unit	**Item**	
1.0	**Cookie**	1105
1.5	**ERROR**	37
	Tea	1023
	UNKNOWN	40
2.0	**Coffee**	1108
	ERROR	31
	UNKNOWN	49
3.0	**Cake**	1085
	ERROR	77
	Juice	1110
	UNKNOWN	77
4.0	**ERROR**	61
	Sandwich	1082
	Smoothie	1036
	UNKNOWN	70
5.0	**ERROR**	39
	Salad	1082
	UNKNOWN	45
ERROR	**Cake**	19
	Coffee	18
	Cookie	21
	ERROR	3
	Juice	26
	Salad	34
	Sandwich	13
	Smoothie	19
	Tea	25
	UNKNOWN	4

		Transaction ID
Price Per Unit	**Item**	
UNKNOWN	**Cake**	14
	Coffee	20
	Cookie	21
	ERROR	3
	Juice	18
	Salad	16
	Sandwich	19
	Smoothie	17
	Tea	21
	UNKNOWN	7

In [112]:
```
mask = (cafe_df['Item'].isin(['ERROR','UNKNOWN'])) & (cafe_df['Price Per Unit'] == '1.5')

cafe_df.loc[mask, 'Item'] = 'Tea'
```

In [113]:
```
mask = (cafe_df['Item'].isin(['ERROR','UNKNOWN'])) & (cafe_df['Price Per Unit'] == '2.0')

cafe_df.loc[mask, 'Item'] = 'Coffee'
```

In [114]:
```
cafe_df.groupby(['Price Per Unit','Item']).agg({
    'Transaction ID':'count'
})
```

Out[114]:

		Transaction ID
Price Per Unit	**Item**	
1.0	**Cookie**	1105
1.5	**Tea**	1100
2.0	**Coffee**	1188
3.0	**Cake**	1085
	ERROR	77
	Juice	1110
	UNKNOWN	77
4.0	**ERROR**	61
	Sandwich	1082
	Smoothie	1036
	UNKNOWN	70
5.0	**ERROR**	39
	Salad	1082
	UNKNOWN	45
ERROR	**Cake**	19
	Coffee	18
	Cookie	21
	ERROR	3
	Juice	26
	Salad	34
	Sandwich	13
	Smoothie	19
	Tea	25
	UNKNOWN	4
UNKNOWN	**Cake**	14
	Coffee	20
	Cookie	21
	ERROR	3
	Juice	18
	Salad	16
	Sandwich	19
	Smoothie	17
	Tea	21
	UNKNOWN	7

In [115]:
```
cake_mask = (cafe_df['Item'].isin(['ERROR'])) & (cafe_df['Price Per Unit'] ==
'3.0')
cafe_df.loc[cake_mask, 'Item'] = 'Cake'

juice_mask = (cafe_df['Item'].isin(['UNKNOWN'])) & (cafe_df['Price Per Unit']
== '3.0')
cafe_df.loc[juice_mask, 'Item'] = 'Juice'
```

In [116]:
```
cafe_df.groupby(['Price Per Unit','Item']).agg({
    'Transaction ID':'count'
})
```

Out[116]:

		Transaction ID
Price Per Unit	**Item**	
1.0	**Cookie**	1105
1.5	**Tea**	1100
2.0	**Coffee**	1188
3.0	**Cake**	1162
	Juice	1187
4.0	**ERROR**	61
	Sandwich	1082
	Smoothie	1036
	UNKNOWN	70
5.0	**ERROR**	39
	Salad	1082
	UNKNOWN	45
ERROR	**Cake**	19
	Coffee	18
	Cookie	21
	ERROR	3
	Juice	26
	Salad	34
	Sandwich	13
	Smoothie	19
	Tea	25
	UNKNOWN	4
UNKNOWN	**Cake**	14
	Coffee	20
	Cookie	21
	ERROR	3
	Juice	18
	Salad	16
	Sandwich	19
	Smoothie	17
	Tea	21
	UNKNOWN	7

In [117]:
```
cake_mask = (cafe_df['Item'].isin(['ERROR'])) & (cafe_df['Price Per Unit'] ==
'4.0')
cafe_df.loc[cake_mask, 'Item'] = 'Sandwich'

juice_mask = (cafe_df['Item'].isin(['UNKNOWN'])) & (cafe_df['Price Per Unit']
== '4.0')
cafe_df.loc[juice_mask, 'Item'] = 'Smoothie'
```

In [118]:
```
cafe_df.groupby(['Price Per Unit','Item']).agg({
    'Transaction ID':'count'
})
```

Out[118]:

		Transaction ID
Price Per Unit	**Item**	
1.0	**Cookie**	1105
1.5	**Tea**	1100
2.0	**Coffee**	1188
3.0	**Cake**	1162
	Juice	1187
4.0	**Sandwich**	1143
	Smoothie	1106
5.0	**ERROR**	39
	Salad	1082
	UNKNOWN	45
ERROR	**Cake**	19
	Coffee	18
	Cookie	21
	ERROR	3
	Juice	26
	Salad	34
	Sandwich	13
	Smoothie	19
	Tea	25
	UNKNOWN	4
UNKNOWN	**Cake**	14
	Coffee	20
	Cookie	21
	ERROR	3
	Juice	18
	Salad	16
	Sandwich	19
	Smoothie	17
	Tea	21
	UNKNOWN	7

In [119]:
```
salad_mask = (cafe_df['Item'].isin(['ERROR','UNKNOWN'])) & (cafe_df['Price Per
Unit'] == '5.0')
cafe_df.loc[salad_mask, 'Item'] = 'Salad'
```

In [120]:
```
# dictionary of correct values
item_price_map = {
    "Cake": '3.0',
    "Tea": '1.5',
    "Coffee": '2.0',
    "Cookie":'1.0',
    "Juice":'3.0',
    "Sandwich":'4.0',
    "Smoothie":'4.0',
    "Salad":'5.0'
}

for item, price in item_price_map.items():
    mask = cafe_df['Price Per Unit'].isin(['ERROR','UNKNOWN']) & (cafe_df['Ite
m'] == item)
    cafe_df.loc[mask, 'Price Per Unit'] = price
```

In [121]:
```
cafe_df.groupby(['Price Per Unit','Item']).agg({
    'Transaction ID':'count'
})
```

Out[121]:

		Transaction ID
Price Per Unit	**Item**	
1.0	**Cookie**	1147
1.5	**Tea**	1146
2.0	**Coffee**	1226
3.0	**Cake**	1195
	Juice	1231
4.0	**Sandwich**	1175
	Smoothie	1142
5.0	**Salad**	1216
ERROR	**ERROR**	3
	UNKNOWN	4
UNKNOWN	**ERROR**	3
	UNKNOWN	7

In [122]:
```
cafe_df = cafe_df[(~cafe_df['Price Per Unit'].isin(['ERROR','UNKNOWN']))&(~caf
e_df['Item'].isin(['ERROR','UNKNOWN']))]
```

In [123]:
```
cafe_df.groupby(['Price Per Unit','Item']).agg({
    'Transaction ID':'count'
})
```

Out[123]:

		Transaction ID
Price Per Unit	**Item**	
1.0	**Cookie**	1147
1.5	**Tea**	1146
2.0	**Coffee**	1226
3.0	**Cake**	1195
	Juice	1231
4.0	**Sandwich**	1175
	Smoothie	1142
5.0	**Salad**	1216

In [124]:
```
cafe_df.to_csv('C:/Users/Mr.Ishan/Downloads/Cafe Sales Dirty/clean_cafe_sales.
csv',index=False)
```

In [125]:
```
cafe_df['Payment Method'].value_counts()
```

Out[125]:
```
Payment Method
Digital Wallet    2285
Credit Card       2258
Cash              2252
ERROR              304
UNKNOWN            292
Name: count, dtype: int64
```

In [126]:
```
cafe_df['Location'].value_counts()
```

Out[126]:
```
Location
Takeaway    3009
In-store    3004
ERROR        357
UNKNOWN      337
Name: count, dtype: int64
```

```
In [127]: cafe_df
```

Out[127]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
0	TXN_1961373	Coffee	2	2.0	4.0	Credit Card	Takeaway	2023-09-08
1	TXN_4977031	Cake	4	3.0	12.0	Cash	In-store	2023-05-16
2	TXN_4271903	Cookie	4	1.0	ERROR	Credit Card	In-store	2023-07-19
3	TXN_7034554	Salad	2	5.0	10.0	UNKNOWN	UNKNOWN	2023-04-27
4	TXN_3160411	Coffee	2	2.0	4.0	Digital Wallet	In-store	2023-06-11
...	...	...	...	...	...	...	...	...
9995	TXN_7672686	Coffee	2	2.0	4.0	NaN	UNKNOWN	2023-08-30
9996	TXN_9659401	NaN	3	NaN	3.0	Digital Wallet	NaN	2023-06-02
9997	TXN_5255387	Coffee	4	2.0	8.0	Digital Wallet	NaN	2023-03-02
9998	TXN_7695629	Cookie	3	NaN	3.0	Digital Wallet	NaN	2023-12-02
9999	TXN_6170729	Sandwich	3	4.0	12.0	Cash	In-store	2023-11-07

9953 rows × 8 columns

```
In [128]: cafe_df.isna().sum()
```

Out[128]:

```
Transaction ID         0
Item                 317
Quantity             138
Price Per Unit       165
Total Spent          172
Payment Method      2562
Location            3246
Transaction Date     157
dtype: int64
```

In [129]:
```
cafe_df[cafe_df['Item'].isna()]
```

Out[129]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
8	TXN_4717867	NaN	5	3.0	15.0	NaN	Takeaway	2023-07-28
30	TXN_1736287	NaN	5	2.0	10.0	Digital Wallet	NaN	2023-06-02
61	TXN_8051289	NaN	1	3.0	3.0	NaN	In-store	2023-10-09
72	TXN_6044979	NaN	1	1.0	1.0	Cash	In-store	2023-12-08
89	TXN_4132730	NaN	5	1.0	5.0	NaN	In-store	2023-03-12
...	...	...	...	...	...	...	...	...
9820	TXN_8751702	NaN	5	NaN	15.0	Cash	NaN	2023-02-13
9855	TXN_3740505	NaN	2	1.5	3.0	NaN	NaN	2023-11-21
9876	TXN_3105633	NaN	1	2.0	2.0	NaN	In-store	2023-03-30
9885	TXN_4659954	NaN	3	4.0	12.0	Credit Card	In-store	NaN
9996	TXN_9659401	NaN	3	NaN	3.0	Digital Wallet	NaN	2023-06-02

317 rows × 8 columns

In [130]:
```
cafe_df[cafe_df['Item'].isna()].groupby('Price Per Unit').agg({
    'Transaction ID':'count'
})
```

Out[130]:

	Transaction ID
Price Per Unit	
1.0	38
1.5	33
2.0	39
3.0	80
4.0	82
5.0	38

In [131]:
```
# Explicit mapping
price_to_item = {
    "1.0": "Cookie",
    "1.5": "Tea",
    "2.0": "Coffee",
    "5.0": "Salad"
}
```

In [132]:
```
# Fill NaN in Item explicitly
cafe_df["Item"] = cafe_df.apply(
    lambda row: price_to_item.get(row["Price Per Unit"], row["Item"]),
    axis=1
)
```

In [133]:
```
cafe_df[cafe_df['Item'].isna()].groupby('Price Per Unit').agg({
    'Transaction ID':'count'
})
```

Out[133]:

	Transaction ID
Price Per Unit	
3.0	80
4.0	82

In [134]:
```
cafe_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 9953 entries, 0 to 9999
Data columns (total 8 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   Transaction ID    9953 non-null   object
 1   Item              9784 non-null   object
 2   Quantity          9815 non-null   object
 3   Price Per Unit    9788 non-null   object
 4   Total Spent       9781 non-null   object
 5   Payment Method    7391 non-null   object
 6   Location          6707 non-null   object
 7   Transaction Date  9796 non-null   object
dtypes: object(8)
memory usage: 699.8+ KB
```

In [135]:
```
multi_mapping = {
    "3.0": ["Cake", "Juice"],
    "4.0": ["Sandwich", "Smoothie"]
}
```

In [136]:
```
# Function to split NaNs half-half
def fill_half_half(group, options):
    na_mask = group["Item"].isna()
    n_missing = na_mask.sum()
    if n_missing == 0:
        return group

    # Create alternating assignments (e.g., Cake, Pie, Cake, Pie, …)
    fills = np.tile(options, int(np.ceil(n_missing / len(options))))[:n_missin
g]

    # Assign back to NaNs
    group.loc[na_mask, "Item"] = fills
    return group

# Apply to each price group that needs splitting
for price, items in multi_mapping.items():
    mask = cafe_df["Price Per Unit"] == price
    cafe_df.loc[mask] = fill_half_half(cafe_df.loc[mask].copy(), items)
```

In [137]:
```
cafe_df.groupby(['Price Per Unit','Item']).agg({
    'Transaction ID':'count'
})
```

Out[137]:

		Transaction ID
Price Per Unit	**Item**	
1.0	**Cookie**	1185
1.5	**Tea**	1179
2.0	**Coffee**	1265
3.0	**Cake**	1235
	Juice	1271
4.0	**Sandwich**	1216
	Smoothie	1183
5.0	**Salad**	1254

In [63]:
```
# The logic was fill the Error Unknown and NaN values of Items with the ones w
hose proce per unit match
```

In [138]:
```
cafe_df.isna().sum()
```

Out[138]:
```
Transaction ID         0
Item                   7
Quantity             138
Price Per Unit       165
Total Spent          172
Payment Method      2562
Location            3246
Transaction Date     157
dtype: int64
```

In [139]:
```
cafe_df[cafe_df['Item'].isna()]
```

Out[139]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
151	TXN_4031509	NaN	4	NaN	16.0	Credit Card	Takeaway	2023-01-04
334	TXN_2523298	NaN	4	NaN	6.0	ERROR	In-store	2023-03-25
818	TXN_7940202	NaN	1	NaN	4.0	Digital Wallet	NaN	2023-07-23
6429	TXN_2536573	NaN	2	NaN	8.0	Cash	In-store	2023-06-24
9819	TXN_1208561	NaN	ERROR	NaN	20.0	Credit Card	NaN	2023-08-19
9820	TXN_8751702	NaN	5	NaN	15.0	Cash	NaN	2023-02-13
9996	TXN_9659401	NaN	3	NaN	3.0	Digital Wallet	NaN	2023-06-02

In [140]:
```
cafe_df['Quantity'].value_counts()
```

Out[140]:
```
Quantity
5          2002
2          1967
4          1850
3          1840
1          1817
ERROR       170
UNKNOWN     169
Name: count, dtype: int64
```

In [141]:
```
quant_mask = (cafe_df['Quantity'].isna())
tot_spent_mask = cafe_df['Total Spent'].isna()
ppu_mask = cafe_df['Price Per Unit'].isna()
```

In [142]:
```
cafe_df[quant_mask & tot_spent_mask]
```

Out[142]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
3401	TXN_3251829	Tea	NaN	1.5	NaN	Digital Wallet	In-store	2023-07-25
8479	TXN_1547245	Sandwich	NaN	4.0	NaN	NaN	Takeaway	2023-09-11
8732	TXN_4550558	Cookie	NaN	1.0	NaN	Credit Card	In-store	2023-08-04

In [143]:
```
cafe_df[quant_mask].isna().sum()
```

Out[143]:
```
Transaction ID        0
Item                  0
Quantity            138
Price Per Unit        1
Total Spent           3
Payment Method       33
Location             41
Transaction Date      0
dtype: int64
```

In [144]:
```
cafe_df[quant_mask & ppu_mask]
```

Out[144]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
912	TXN_1575608	Sandwich	NaN	NaN	20.0	ERROR	Takeaway	2023-01-05

In [145]:
```
cafe_df[quant_mask]
```

Out[145]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
66	TXN_8501819	Juice	NaN	3.0	6.0	Cash	NaN	2023-03-30
341	TXN_2265316	Cookie	NaN	1.0	3.0	Credit Card	In-store	2023-12-29
376	TXN_6319728	Coffee	NaN	2.0	4.0	Credit Card	In-store	2023-07-18
412	TXN_4660753	Juice	NaN	3.0	3.0	Credit Card	Takeaway	2023-10-04
532	TXN_7533411	Cookie	NaN	1.0	1.0	Digital Wallet	In-store	2023-11-09
...	...	...	...	...	...	...	...	...
9634	TXN_8436045	Cake	NaN	3.0	15.0	Credit Card	NaN	2023-08-08
9844	TXN_4528914	Salad	NaN	5.0	5.0	ERROR	In-store	2023-08-06
9869	TXN_1975184	Coffee	NaN	2.0	UNKNOWN	Digital Wallet	NaN	2023-01-15
9887	TXN_8963470	Salad	NaN	5.0	10.0	NaN	In-store	2023-06-01
9896	TXN_9089045	Coffee	NaN	2.0	10.0	Cash	In-store	2023-03-12

138 rows × 8 columns

In [146]:
```
cafe_df = cafe_df.replace(["ERROR", "UNKNOWN"], np.nan)
```

In [147]:
```
cafe_df[quant_mask]
```

Out[147]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
66	TXN_8501819	Juice	NaN	3.0	6.0	Cash	NaN	2023-03-30
341	TXN_2265316	Cookie	NaN	1.0	3.0	Credit Card	In-store	2023-12-29
376	TXN_6319728	Coffee	NaN	2.0	4.0	Credit Card	In-store	2023-07-18
412	TXN_4660753	Juice	NaN	3.0	3.0	Credit Card	Takeaway	2023-10-04
532	TXN_7533411	Cookie	NaN	1.0	1.0	Digital Wallet	In-store	2023-11-09
...	...	...	...	...	...	...	...	...
9634	TXN_8436045	Cake	NaN	3.0	15.0	Credit Card	NaN	2023-08-08
9844	TXN_4528914	Salad	NaN	5.0	5.0	NaN	In-store	2023-08-06
9869	TXN_1975184	Coffee	NaN	2.0	NaN	Digital Wallet	NaN	2023-01-15
9887	TXN_8963470	Salad	NaN	5.0	10.0	NaN	In-store	2023-06-01
9896	TXN_9089045	Coffee	NaN	2.0	10.0	Cash	In-store	2023-03-12

138 rows × 8 columns

In [148]:
```
# Step 2: Convert columns to numeric (non-numeric become NaN automatically)
cafe_df["Quantity"] = pd.to_numeric(cafe_df["Quantity"], errors="coerce")
cafe_df["Price Per Unit"] = pd.to_numeric(cafe_df["Price Per Unit"], errors="c
oerce")
cafe_df["Total Spent"] = pd.to_numeric(cafe_df["Total Spent"], errors="coerc
e")
```

In [149]:
```
# Step 3: Where Quantity is missing, but Price & Total are valid → calculate
mask = cafe_df["Quantity"].isna() & cafe_df["Price Per Unit"].notna() & cafe_d
f["Total Spent"].notna()
cafe_df.loc[mask, "Quantity"] = cafe_df.loc[mask, "Total Spent"] / cafe_df.loc
[mask, "Price Per Unit"]
```

In [150]:
```
cafe_df.isna().sum()
```

Out[150]:
```
Transaction ID         0
Item                   7
Quantity              26
Price Per Unit       165
Total Spent          499
Payment Method      3158
Location            3940
Transaction Date     457
dtype: int64
```

In [151]:
```
cafe_df.isna().sum()
```

Out[151]:
```
Transaction ID         0
Item                   7
Quantity              26
Price Per Unit       165
Total Spent          499
Payment Method      3158
Location            3940
Transaction Date     457
dtype: int64
```

In [152]:
```
mask = cafe_df["Price Per Unit"].isna() & cafe_df["Quantity"].notna() & cafe_d
f["Total Spent"].notna()
cafe_df.loc[mask, "Price Per Unit"] = cafe_df.loc[mask, "Total Spent"] / cafe_
df.loc[mask, "Quantity"]
```

In [153]:
```
cafe_df.isna().sum()
```

Out[153]:
```
Transaction ID         0
Item                   7
Quantity              26
Price Per Unit        11
Total Spent          499
Payment Method      3158
Location            3940
Transaction Date     457
dtype: int64
```

In [154]:
```
mask = cafe_df["Total Spent"].isna() & cafe_df["Quantity"].notna() & cafe_df
["Price Per Unit"].notna()
cafe_df.loc[mask, "Total Spent"] = cafe_df.loc[mask, "Price Per Unit"] * cafe_
df.loc[mask, "Quantity"]
```

In [155]:
```
cafe_df.isna().sum()
```

Out[155]:
```
Transaction ID         0
Item                   7
Quantity              26
Price Per Unit        11
Total Spent           25
Payment Method      3158
Location            3940
Transaction Date     457
dtype: int64
```

```
In [156]: cafe_df[(cafe_df['Item'].isna())|(cafe_df['Quantity'].isna())|(cafe_df['Price
          Per Unit'].isna())]
```

Out[156]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
65	TXN_4987129	Sandwich	3.0	NaN	NaN	NaN	In-store	2023-10-20
151	TXN_4031509	NaN	4.0	4.0	16.0	Credit Card	Takeaway	2023-01-04
236	TXN_8562645	Salad	NaN	5.0	NaN	NaN	In-store	2023-05-18
278	TXN_3229409	Juice	NaN	3.0	NaN	Cash	Takeaway	2023-04-15
334	TXN_2523298	NaN	4.0	1.5	6.0	NaN	In-store	2023-03-25
629	TXN_9289174	Cake	NaN	NaN	12.0	Digital Wallet	In-store	2023-12-30
641	TXN_2962976	Juice	NaN	3.0	NaN	NaN	NaN	2023-03-17
738	TXN_8696094	Sandwich	NaN	4.0	NaN	NaN	Takeaway	2023-05-14
818	TXN_7940202	NaN	1.0	4.0	4.0	Digital Wallet	NaN	2023-07-23
912	TXN_1575608	Sandwich	NaN	NaN	20.0	NaN	Takeaway	2023-01-05
1008	TXN_7225428	Tea	NaN	NaN	3.0	Credit Card	Takeaway	2023-03-07
1482	TXN_3593060	Smoothie	NaN	NaN	16.0	Cash	NaN	2023-03-05
1674	TXN_9367492	Tea	2.0	NaN	NaN	Cash	In-store	2023-06-19
2796	TXN_9188692	Cake	NaN	3.0	NaN	Credit Card	NaN	2023-12-01
3162	TXN_3577949	Cake	3.0	NaN	NaN	NaN	Takeaway	2023-04-25
3203	TXN_4565754	Smoothie	NaN	4.0	NaN	Digital Wallet	Takeaway	2023-10-06
3224	TXN_6297232	Coffee	NaN	2.0	NaN	NaN	NaN	2023-04-07
3401	TXN_3251829	Tea	NaN	1.5	NaN	Digital Wallet	In-store	2023-07-25
4257	TXN_6470865	Coffee	NaN	2.0	NaN	Digital Wallet	Takeaway	2023-09-18
5841	TXN_5884081	Cookie	NaN	1.0	NaN	Digital Wallet	In-store	2023-07-05
6225	TXN_6859249	Cookie	NaN	NaN	2.0	NaN	NaN	NaN
6429	TXN_2536573	NaN	2.0	4.0	8.0	Cash	In-store	2023-06-24
7029	TXN_4628338	Coffee	NaN	2.0	NaN	Cash	NaN	2023-12-25
7035	TXN_8872984	Salad	5.0	NaN	NaN	Credit Card	In-store	2023-08-23
7297	TXN_9944500	Smoothie	NaN	4.0	NaN	Cash	In-store	2023-01-03
8021	TXN_2428781	Salad	NaN	5.0	NaN	NaN	In-store	2023-05-09
8443	TXN_2023651	Sandwich	NaN	4.0	NaN	Cash	In-store	2023-05-25
8465	TXN_9669616	Coffee	NaN	2.0	NaN	NaN	NaN	2023-06-03
8479	TXN_1547245	Sandwich	NaN	4.0	NaN	NaN	Takeaway	2023-09-11
8574	TXN_2546684	Juice	NaN	3.0	NaN	Digital Wallet	Takeaway	2023-04-08
8732	TXN_4550558	Cookie	NaN	1.0	NaN	Credit Card	In-store	2023-08-04

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
9590	TXN_9924732	Sandwich	NaN	4.0	NaN	Credit Card	In-store	2023-01-18
9819	TXN_1208561	NaN	NaN	NaN	20.0	Credit Card	NaN	2023-08-19
9820	TXN_8751702	NaN	5.0	3.0	15.0	Cash	NaN	2023-02-13
9869	TXN_1975184	Coffee	NaN	2.0	NaN	Digital Wallet	NaN	2023-01-15
9893	TXN_3809533	Juice	2.0	NaN	NaN	Digital Wallet	Takeaway	2023-02-02
9996	TXN_9659401	NaN	3.0	1.0	3.0	Digital Wallet	NaN	2023-06-02

In [165]:
```
cafe_df[(~cafe_df['Item'].isna()) & (cafe_df['Price Per Unit'].isna())]
```
Out[165]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
65	TXN_4987129	Sandwich	3.0	NaN	NaN	NaN	In-store	2023-10-20
629	TXN_9289174	Cake	NaN	NaN	12.0	Digital Wallet	In-store	2023-12-30
912	TXN_1575608	Sandwich	NaN	NaN	20.0	NaN	Takeaway	2023-01-05
1008	TXN_7225428	Tea	NaN	NaN	3.0	Credit Card	Takeaway	2023-03-07
1482	TXN_3593060	Smoothie	NaN	NaN	16.0	Cash	NaN	2023-03-05
1674	TXN_9367492	Tea	2.0	NaN	NaN	Cash	In-store	2023-06-19
3162	TXN_3577949	Cake	3.0	NaN	NaN	NaN	Takeaway	2023-04-25
6225	TXN_6859249	Cookie	NaN	NaN	2.0	NaN	NaN	NaN
7035	TXN_8872984	Salad	5.0	NaN	NaN	Credit Card	In-store	2023-08-23
9893	TXN_3809533	Juice	2.0	NaN	NaN	Digital Wallet	Takeaway	2023-02-02

In [166]:
```
cafe_df[(cafe_df['Item'].isna()) & (~cafe_df['Price Per Unit'].isna())]
```
Out[166]:

Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date

In [162]:
```
cafe_df[cafe_df['Price Per Unit'] == 4].groupby('Item').count()
```
Out[162]:

	Transaction ID	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
Item							
Sandwich	1231	1227	1231	1227	839	770	1169
Smoothie	1206	1204	1206	1204	822	693	1153

```
In [168]: price_to_item = {
              1.0: "Cookie",
              1.5: "Tea",
              2.0: "Coffee",
              3.0: "Juice",
              4.0: "Sandwich",
              5.0: "Salad"

          }
```

```
In [180]: item_to_price={
              "Cookie":1.0,
              "Tea":1.5,
              "Coffee":2.0,
              "Juice":3.0,
              "Cake":3.0,
              "Sandwich":4.0,
              "Smoothie":4.0,
              "Salad":5.0
          }
```

```
In [174]: # Fill NaN in Item explicitly
          cafe_df["Item"] = cafe_df.apply(
              lambda row: item_to_price.get(row["Price Per Unit"], row["Item"]),
              axis=1
          )
```

```
In [181]: # Fill NaN in Item explicitly
          cafe_df["Price Per Unit"] = cafe_df.apply(
              lambda row: item_to_price.get(row["Item"], row["Price Per Unit"]),
              axis=1
          )
```

```
In [173]: cafe_df.isna().sum()

Out[173]: Transaction ID         0
          Item                   1
          Quantity              26
          Price Per Unit        11
          Total Spent           25
          Payment Method      3158
          Location            3940
          Transaction Date     457
          dtype: int64
```

In [177]:
```
cafe_df.isna().sum()
```

Out[177]:
```
Transaction ID         0
Item                   1
Quantity              26
Price Per Unit         4
Total Spent           25
Payment Method      3158
Location            3940
Transaction Date     457
dtype: int64
```

In [182]:
```
cafe_df[(cafe_df['Item'].isna()) | (cafe_df['Price Per Unit'].isna())]
```

Out[182]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
9819	TXN_1208561	NaN	NaN	NaN	20.0	Credit Card	NaN	2023-08-19

In [183]:
```
cafe_df.isna().sum()
```

Out[183]:
```
Transaction ID         0
Item                   1
Quantity              26
Price Per Unit         1
Total Spent           25
Payment Method      3158
Location            3940
Transaction Date     457
dtype: int64
```

In [187]:
```
# Step 3: Where Quantity is missing, but Price & Total are valid → calculate
mask = cafe_df["Quantity"].isna() & cafe_df["Price Per Unit"].notna() & cafe_d
f["Total Spent"].notna()
cafe_df.loc[mask, "Quantity"] = cafe_df.loc[mask, "Total Spent"] / cafe_df.loc
[mask, "Price Per Unit"]
```

In [188]:
```
mask = cafe_df["Price Per Unit"].isna() & cafe_df["Quantity"].notna() & cafe_d
f["Total Spent"].notna()
cafe_df.loc[mask, "Price Per Unit"] = cafe_df.loc[mask, "Total Spent"] / cafe_
df.loc[mask, "Quantity"]
```

In [189]:
```
cafe_df.isna().sum()
```

Out[189]:
```
Transaction ID         0
Item                   1
Quantity              21
Price Per Unit         1
Total Spent           25
Payment Method      3158
Location            3940
Transaction Date     457
dtype: int64
```

In [191]:
```
cafe_df[cafe_df['Total Spent'].isna()]
```

Out[191]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
65	TXN_4987129	Sandwich	3.0	4.0	NaN	NaN	In-store	2023-10-20
236	TXN_8562645	Salad	NaN	5.0	NaN	NaN	In-store	2023-05-18
278	TXN_3229409	Juice	NaN	3.0	NaN	Cash	Takeaway	2023-04-15
641	TXN_2962976	Juice	NaN	3.0	NaN	NaN	NaN	2023-03-17
738	TXN_8696094	Sandwich	NaN	4.0	NaN	NaN	Takeaway	2023-05-14
1674	TXN_9367492	Tea	2.0	1.5	NaN	Cash	In-store	2023-06-19
2796	TXN_9188692	Juice	NaN	3.0	NaN	Credit Card	NaN	2023-12-01
3162	TXN_3577949	Cake	3.0	3.0	NaN	NaN	Takeaway	2023-04-25
3203	TXN_4565754	Sandwich	NaN	4.0	NaN	Digital Wallet	Takeaway	2023-10-06
3224	TXN_6297232	Coffee	NaN	2.0	NaN	NaN	NaN	2023-04-07
3401	TXN_3251829	Tea	NaN	1.5	NaN	Digital Wallet	In-store	2023-07-25
4257	TXN_6470865	Coffee	NaN	2.0	NaN	Digital Wallet	Takeaway	2023-09-18
5841	TXN_5884081	Cookie	NaN	1.0	NaN	Digital Wallet	In-store	2023-07-05
7029	TXN_4628338	Coffee	NaN	2.0	NaN	Cash	NaN	2023-12-25
7035	TXN_8872984	Salad	5.0	5.0	NaN	Credit Card	In-store	2023-08-23
7297	TXN_9944500	Sandwich	NaN	4.0	NaN	Cash	In-store	2023-01-03
8021	TXN_2428781	Salad	NaN	5.0	NaN	NaN	In-store	2023-05-09
8443	TXN_2023651	Sandwich	NaN	4.0	NaN	Cash	In-store	2023-05-25
8465	TXN_9669616	Coffee	NaN	2.0	NaN	NaN	NaN	2023-06-03
8479	TXN_1547245	Sandwich	NaN	4.0	NaN	NaN	Takeaway	2023-09-11
8574	TXN_2546684	Juice	NaN	3.0	NaN	Digital Wallet	Takeaway	2023-04-08
8732	TXN_4550558	Cookie	NaN	1.0	NaN	Credit Card	In-store	2023-08-04
9590	TXN_9924732	Sandwich	NaN	4.0	NaN	Credit Card	In-store	2023-01-18
9869	TXN_1975184	Coffee	NaN	2.0	NaN	Digital Wallet	NaN	2023-01-15
9893	TXN_3809533	Juice	2.0	3.0	NaN	Digital Wallet	Takeaway	2023-02-02

In [192]:
```
mask = cafe_df["Total Spent"].isna() & cafe_df["Quantity"].notna() & cafe_df
["Price Per Unit"].notna()
cafe_df.loc[mask, "Total Spent"] = cafe_df.loc[mask, "Price Per Unit"] * cafe_
df.loc[mask, "Quantity"]
```

In [193]:
```
cafe_df.isna().sum()
```

Out[193]:
```
Transaction ID         0
Item                   1
Quantity              21
Price Per Unit         1
Total Spent           20
Payment Method      3158
Location            3940
Transaction Date     457
dtype: int64
```

In [194]:
```
# Step 2: Fill NaNs in Quantity with mode per Item
cafe_df["Quantity"] = cafe_df.groupby("Item")["Quantity"].transform(
    lambda x: x.fillna(x.mode().iloc[0] if not x.mode().empty else np.nan)
)
```

In [195]:
```
cafe_df.isna().sum()
```

Out[195]:
```
Transaction ID         0
Item                   1
Quantity               1
Price Per Unit         1
Total Spent           20
Payment Method      3158
Location            3940
Transaction Date     457
dtype: int64
```

In [196]:
```
cafe_df
```

Out[196]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
0	TXN_1961373	Coffee	2.0	2.0	4.0	Credit Card	Takeaway	2023-09-08
1	TXN_4977031	Juice	4.0	3.0	12.0	Cash	In-store	2023-05-16
2	TXN_4271903	Cookie	4.0	1.0	4.0	Credit Card	In-store	2023-07-19
3	TXN_7034554	Salad	2.0	5.0	10.0	NaN	NaN	2023-04-27
4	TXN_3160411	Coffee	2.0	2.0	4.0	Digital Wallet	In-store	2023-06-11
...	...	...	...	...	...	...	...	...
9995	TXN_7672686	Coffee	2.0	2.0	4.0	NaN	NaN	2023-08-30
9996	TXN_9659401	Cookie	3.0	1.0	3.0	Digital Wallet	NaN	2023-06-02
9997	TXN_5255387	Coffee	4.0	2.0	8.0	Digital Wallet	NaN	2023-03-02
9998	TXN_7695629	Cookie	3.0	1.0	3.0	Digital Wallet	NaN	2023-12-02
9999	TXN_6170729	Sandwich	3.0	4.0	12.0	Cash	In-store	2023-11-07

9953 rows × 8 columns

In [197]:
```
mask = cafe_df["Total Spent"].isna() & cafe_df["Quantity"].notna() & cafe_df
["Price Per Unit"].notna()
cafe_df.loc[mask, "Total Spent"] = cafe_df.loc[mask, "Price Per Unit"] * cafe_
df.loc[mask, "Quantity"]
```

In [198]:
```
cafe_df.isna().sum()
```

Out[198]:
```
Transaction ID         0
Item                   1
Quantity               1
Price Per Unit         1
Total Spent            0
Payment Method      3158
Location            3940
Transaction Date     457
dtype: int64
```

In [205]:
```
cafe_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 9953 entries, 0 to 9999
Data columns (total 8 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Transaction ID    9953 non-null   object 
 1   Item              9952 non-null   object 
 2   Quantity          9952 non-null   float64
 3   Price Per Unit    9952 non-null   float64
 4   Total Spent       9953 non-null   float64
 5   Payment Method    6795 non-null   object 
 6   Location          6013 non-null   object 
 7   Transaction Date  9496 non-null   object 
dtypes: float64(3), object(5)
memory usage: 699.8+ KB
```

In [206]:
```
cafe_df['Transaction Date'] = pd.to_datetime(cafe_df['Transaction Date'])
```

In [207]:
```
cafe_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 9953 entries, 0 to 9999
Data columns (total 8 columns):
 #   Column            Non-Null Count  Dtype         
---  ------            --------------  -----         
 0   Transaction ID    9953 non-null   object        
 1   Item              9952 non-null   object        
 2   Quantity          9952 non-null   float64       
 3   Price Per Unit    9952 non-null   float64       
 4   Total Spent       9953 non-null   float64       
 5   Payment Method    6795 non-null   object        
 6   Location          6013 non-null   object        
 7   Transaction Date  9496 non-null   datetime64[ns]
dtypes: datetime64[ns](1), float64(3), object(4)
memory usage: 699.8+ KB
```

In [208]:
```
cafe_df[cafe_df['Transaction Date'].isna()]
```
Out[208]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
11	TXN_3051279	Sandwich	2.0	4.0	8.0	Credit Card	Takeaway	NaT
29	TXN_7640952	Juice	4.0	3.0	12.0	Digital Wallet	Takeaway	NaT
33	TXN_7710508	Cookie	5.0	1.0	5.0	Cash	NaN	NaT
77	TXN_2091733	Salad	1.0	5.0	5.0	NaN	In-store	NaT
103	TXN_7028009	Juice	4.0	3.0	12.0	NaN	Takeaway	NaT
...	...	...	...	...	...	...	...	...
9933	TXN_9460419	Juice	1.0	3.0	3.0	NaN	Takeaway	NaT
9937	TXN_8253472	Juice	1.0	3.0	3.0	NaN	NaN	NaT
9949	TXN_3130865	Juice	3.0	3.0	9.0	NaN	In-store	NaT
9983	TXN_9226047	Sandwich	3.0	4.0	12.0	Cash	NaN	NaT
9988	TXN_9594133	Juice	5.0	3.0	15.0	NaN	NaN	NaT

457 rows × 8 columns

In [210]:
```
cafe_df[:13]
```
Out[210]:

	Transaction ID	Item	Quantity	Price Per Unit	Total Spent	Payment Method	Location	Transaction Date
0	TXN_1961373	Coffee	2.0	2.0	4.0	Credit Card	Takeaway	2023-09-08
1	TXN_4977031	Juice	4.0	3.0	12.0	Cash	In-store	2023-05-16
2	TXN_4271903	Cookie	4.0	1.0	4.0	Credit Card	In-store	2023-07-19
3	TXN_7034554	Salad	2.0	5.0	10.0	NaN	NaN	2023-04-27
4	TXN_3160411	Coffee	2.0	2.0	4.0	Digital Wallet	In-store	2023-06-11
5	TXN_2602893	Sandwich	5.0	4.0	20.0	Credit Card	NaN	2023-03-31
6	TXN_4433211	Juice	3.0	3.0	9.0	NaN	Takeaway	2023-10-06
7	TXN_6699534	Sandwich	4.0	4.0	16.0	Cash	NaN	2023-10-28
8	TXN_4717867	Juice	5.0	3.0	15.0	NaN	Takeaway	2023-07-28
9	TXN_2064365	Sandwich	5.0	4.0	20.0	NaN	In-store	2023-12-31
10	TXN_2548360	Salad	5.0	5.0	25.0	Cash	Takeaway	2023-11-07
11	TXN_3051279	Sandwich	2.0	4.0	8.0	Credit Card	Takeaway	NaT
12	TXN_7619095	Sandwich	2.0	4.0	8.0	Cash	In-store	2023-05-03

In [215]:
```
cafe_df['Month Name'] =cafe_df['Transaction Date'].dt.month_name()
cafe_df['Year'] =cafe_df['Transaction Date'].dt.year
```

In [219]:
```
cafe_df.groupby(['Year','Month Name']).agg({
    'Transaction ID':'count'
}).sort_values('Transaction ID',ascending=False)
```

Out[219]:

Year	Month Name	Transaction ID
2023.0	October	835
	March	822
	January	818
	June	816
	August	803
	December	787
	September	785
	July	783
	November	780
	April	773
	May	770
	February	724

In [220]:
```
all_dates_2023 = pd.date_range(start="2023-01-01", end="2023-12-31")

# Step 3: Fill NaN with random dates from 2023
na_mask = cafe_df["Transaction Date"].isna()
cafe_df.loc[na_mask, "Transaction Date"] = np.random.choice(all_dates_2023, si
ze=na_mask.sum())
```

In [224]:
```
cafe_df = cafe_df.drop(columns=['Month Name','Year'])
```

In [226]:
```
cafe_df.isna().sum()
```

Out[226]:
```
Transaction ID         0
Item                   1
Quantity               1
Price Per Unit         1
Total Spent            0
Payment Method      3158
Location            3940
Transaction Date       0
dtype: int64
```

In [229]:
```
cafe_df['Payment Method'].value_counts()
```

Out[229]:
```
Payment Method
Digital Wallet    2285
Credit Card       2258
Cash              2252
Name: count, dtype: int64
```

In [230]:
```
all_dates_2023
```

Out[230]:
```
DatetimeIndex(['2023-01-01', '2023-01-02', '2023-01-03', '2023-01-04',
               '2023-01-05', '2023-01-06', '2023-01-07', '2023-01-08',
               '2023-01-09', '2023-01-10',
               ...
               '2023-12-22', '2023-12-23', '2023-12-24', '2023-12-25',
               '2023-12-26', '2023-12-27', '2023-12-28', '2023-12-29',
               '2023-12-30', '2023-12-31'],
              dtype='datetime64[ns]', length=365, freq='D')
```

In [232]:
```
mask = cafe_df["Payment Method"].isna()
cafe_df.loc[mask, "Payment Method"] = np.random.choice(
    ["Credit Card", "Digital Wallet"],
    size=mask.sum())
```

In [233]:
```
cafe_df.isna().sum()
```

Out[233]:
```
Transaction ID         0
Item                   1
Quantity               1
Price Per Unit         1
Total Spent            0
Payment Method         0
Location            3940
Transaction Date       0
dtype: int64
```

In [234]:
```
cafe_df.groupby(['Location']).agg({
    'Transaction ID':'count'
}).sort_values('Transaction ID',ascending=False)
```

Out[234]:

	Transaction ID
Location	
Takeaway	3009
In-store	3004

In [235]:
```
mask = cafe_df["Location"].isna()
cafe_df.loc[mask, "Location"] = np.random.choice(
    ["Takeaway", "In-store"],
    size=mask.sum())
```

In [236]:
```
cafe_df.isna().sum()
```

Out[236]:
```
Transaction ID      0
Item                1
Quantity            1
Price Per Unit      1
Total Spent         0
Payment Method      0
Location            0
Transaction Date    0
dtype: int64
```

In [239]:
```
cafe_df.to_csv('C:/Users/Mr.Ishan/Downloads/Cafe Sales Dirty/clean_cafe_sales.
csv',index=False)
```

In [238]:
```
cafe_df.duplicated().sum()
```

Out[238]:
```
np.int64(0)
```

In [ ]:
```

```