1. Load dataset

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as mp
import seaborn as sb
import re
from datetime import datetime, timedelta
import os

file_path = r"C:\Users\SD\Downloads\Quantium_Experience"

trans = pd.read_excel(file_path + "\\QVI_transaction_data.xlsx")
cus = pd.read_csv(file_path + "\\QVI_purchase_behaviour.csv",encoding='ISO-8859-1')

trans.info()
print(trans.head())

cus.info()
print(cus.head())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 264836 entries, 0 to 264835
Data columns (total 8 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   DATE            264836 non-null  int64
 1   STORE_NBR       264836 non-null  int64
 2   LYLTY_CARD_NBR  264836 non-null  int64
 3   TXN_ID          264836 non-null  int64
 4   PROD_NBR        264836 non-null  int64
 5   PROD_NAME       264836 non-null  object
 6   PROD_QTY        264836 non-null  int64
 7   TOT_SALES       264836 non-null  float64
dtypes: float64(1), int64(6), object(1)
memory usage: 16.2+ MB
    DATE  STORE_NBR  LYLTY_CARD_NBR  TXN_ID  PROD_NBR  \
0  43390          1            1000       1         5
1  43599          1            1307     348        66
2  43605          1            1343     383        61
3  43329          2            2373     974        69
4  43330          2            2426    1038       108

                                PROD_NAME  PROD_QTY  TOT_SALES
0    Natural Chip        Compny SeaSalt175g         2        6.0
1                  CCs Nacho Cheese    175g         3        6.3
2    Smiths Crinkle Cut  Chips Chicken 170g         2        2.9
3    Smiths Chip Thinly  S/Cream&Onion 175g         5       15.0
4  Kettle Tortilla ChpsHny&Jlpno Chili 150g         3       13.8
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 72637 entries, 0 to 72636
Data columns (total 3 columns):
```

```
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   LYLTY_CARD_NBR    72637 non-null  int64
 1   LIFESTAGE         72637 non-null  object
 2   PREMIUM_CUSTOMER  72637 non-null  object
dtypes: int64(1), object(2)
memory usage: 1.7+ MB
   LYLTY_CARD_NBR               LIFESTAGE PREMIUM_CUSTOMER
0            1000   YOUNG SINGLES/COUPLES          Premium
1            1002   YOUNG SINGLES/COUPLES       Mainstream
2            1003          YOUNG FAMILIES           Budget
3            1004   OLDER SINGLES/COUPLES       Mainstream
4            1005  MIDAGE SINGLES/COUPLES       Mainstream
```

2. Clean, prepare and merge datasets

```
print("Transaction missing values: ",trans.isnull().sum())
print("Customer missing values: ",cus.isnull().sum())
```

```
Transaction missing values:  DATE              0
STORE_NBR         0
LYLTY_CARD_NBR    0
TXN_ID            0
PROD_NBR          0
PROD_NAME         0
PROD_QTY          0
TOT_SALES         0
dtype: int64
Customer missing values:  LYLTY_CARD_NBR      0
LIFESTAGE           0
PREMIUM_CUSTOMER    0
dtype: int64
```

```
trans.dropna(inplace=True)
cus.dropna(inplace=True)
```

```
print("Transaction duplicates: ",trans.duplicated().sum())
print("Customer duplicates: ",cus.duplicated().sum())
```

```
Transaction duplicates:  1
Customer duplicates:  0
```

```
trans.drop_duplicates()
cus.drop_duplicates()
```

	LYLTY_CARD_NBR	LIFESTAGE	PREMIUM_CUSTOMER
0	1000	YOUNG SINGLES/COUPLES	Premium
1	1002	YOUNG SINGLES/COUPLES	Mainstream
2	1003	YOUNG FAMILIES	Budget
3	1004	OLDER SINGLES/COUPLES	Mainstream
4	1005	MIDAGE SINGLES/COUPLES	Mainstream
...	...	...	...
72632	2370651	MIDAGE SINGLES/COUPLES	Mainstream
72633	2370701	YOUNG FAMILIES	Mainstream
72634	2370751	YOUNG FAMILIES	Premium
72635	2370961	OLDER FAMILIES	Budget
72636	2373711	YOUNG SINGLES/COUPLES	Mainstream

72637 rows × 3 columns

```
print("Maximun sales:", trans['PROD_QTY'].max())
print("Minimum sales:", trans['PROD_QTY'].min())
trans = trans[(trans['PROD_QTY']>0) & (trans['TOT_SALES']>0)]

trans['PROD_NAME'] = trans['PROD_NAME'].str.strip()
```

```
Maximun sales: 200
Minimum sales: 1
```

```
trans['DATE'] = pd.to_datetime(trans['DATE'], origin='1899-12-30', unit='D')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[90], line 1
----> 1 trans['DATE'] = pd.to_datetime(trans['DATE'], origin='1899-12-30', unit='D')

File ~\anaconda3\Lib\site-packages\pandas\core\tools\datetimes.py:1041, in
to_datetime(arg, errors, dayfirst, yearfirst, utc, format, exact, unit,
infer_datetime_format, origin, cache)
   1038     return None
   1040 if origin != "unix":
-> 1041     arg = _adjust_to_origin(arg, origin, unit)
   1043 convert_listlike = partial(
   1044     _convert_listlike_datetimes,
   1045     utc=utc,
   (...)
   1050     exact=exact,
   1051 )
   1052 # pylint: disable-next=used-before-assignment

File ~\anaconda3\Lib\site-packages\pandas\core\tools\datetimes.py:592, in
_adjust_to_origin(arg, origin, unit)
    587 else:
    588     # arg must be numeric
    589     if not (
    590         (is_integer(arg) or is_float(arg)) or is_numeric_dtype(np.asarray(arg))
    591     ):
--> 592         raise ValueError(
    593             f"'{arg}' is not compatible with origin='{origin}'; "
    594             "it must be numeric with a unit specified"
    595         )
    597     # we are going to offset back to unix / epoch time
    598     try:

ValueError: '0        2018-10-17
1        2019-05-14
2        2019-05-20
3        2018-08-17
4        2018-08-18
           ...
264831   2019-03-09
264832   2018-08-13
264833   2018-11-06
264834   2018-12-27
264835   2018-09-22
Name: DATE, Length: 264836, dtype: datetime64[ns]' is not compatible with origin='1899-
12-30'; it must be numeric with a unit specified
```

```
merged = pd.merge(trans, cus, on='LYLTY_CARD_NBR', how='left')
print("Missing values after merging:",merged.isnull().sum())
```

```
Missing values after merging: DATE              0
STORE_NBR           0
LYLTY_CARD_NBR      0
TXN_ID              0
PROD_NBR            0
PROD_NAME           0
PROD_QTY            0
TOT_SALES           0
LIFESTAGE           0
PREMIUM_CUSTOMER    0
dtype: int64
```

3. Data Analysis

```
merged.head()
```

	DATE	STORE_NBR	LYLTY_CARD_NBR	TXN_ID	PROD_NBR	PROD_NAME	PROD_QTY	TOT_SALE
0	2018-10-17	1	1000	1	5	Natural Chip Compny SeaSalt175g	2	6.
1	2019-05-14	1	1307	348	66	CCs Nacho Cheese 175g	3	6.
2	2019-05-20	1	1343	383	61	Smiths Crinkle Cut Chips Chicken 170g	2	2.
3	2018-08-17	2	2373	974	69	Smiths Chip Thinly S/Cream&Onion 175g	5	15.
4	2018-08-18	2	2426	1038	108	Kettle Tortilla ChpsHny&Jlpno Chili 150g	3	13.

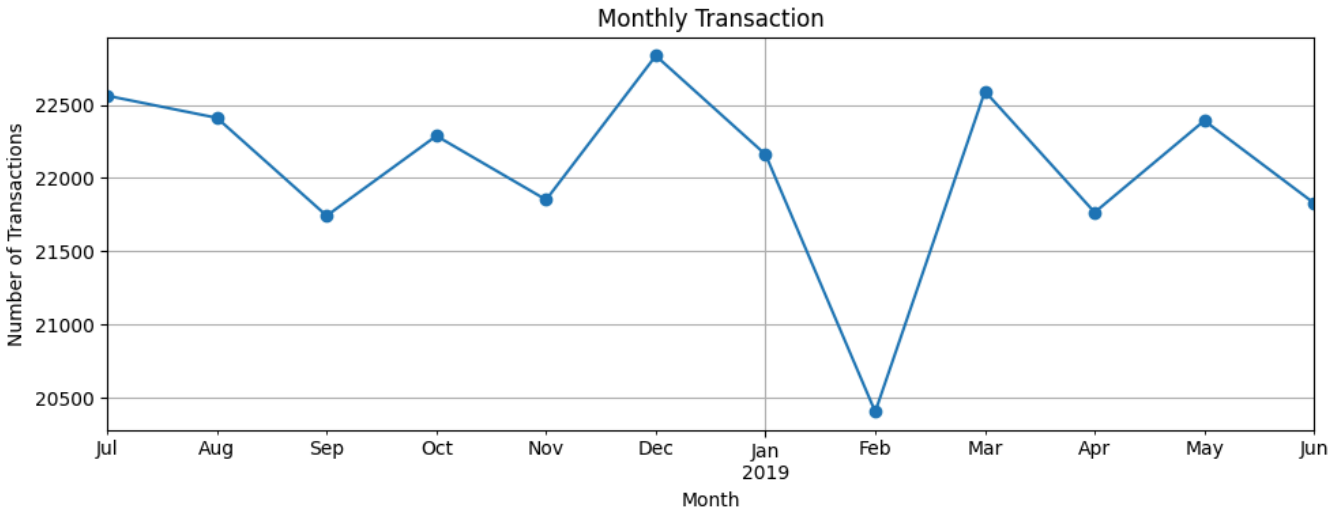
```
print("Transaction Date Range: ",merged['DATE'].min(),"to",merged['DATE'].max())
```

```
Transaction Date Range:  2018-07-01 00:00:00 to 2019-06-30 00:00:00
```

```
monthly_sales = merged.copy()
monthly_sales['MONTH'] = monthly_sales['DATE'].dt.to_period('M')
monthly_counts = monthly_sales.groupby('MONTH').size()

monthly_counts.plot(kind = 'line', marker='o', figsize=(10,4), title='Monthly Transaction')
mp.xlabel('Month')
mp.ylabel('Number of Transactions')
mp.grid(True)
```

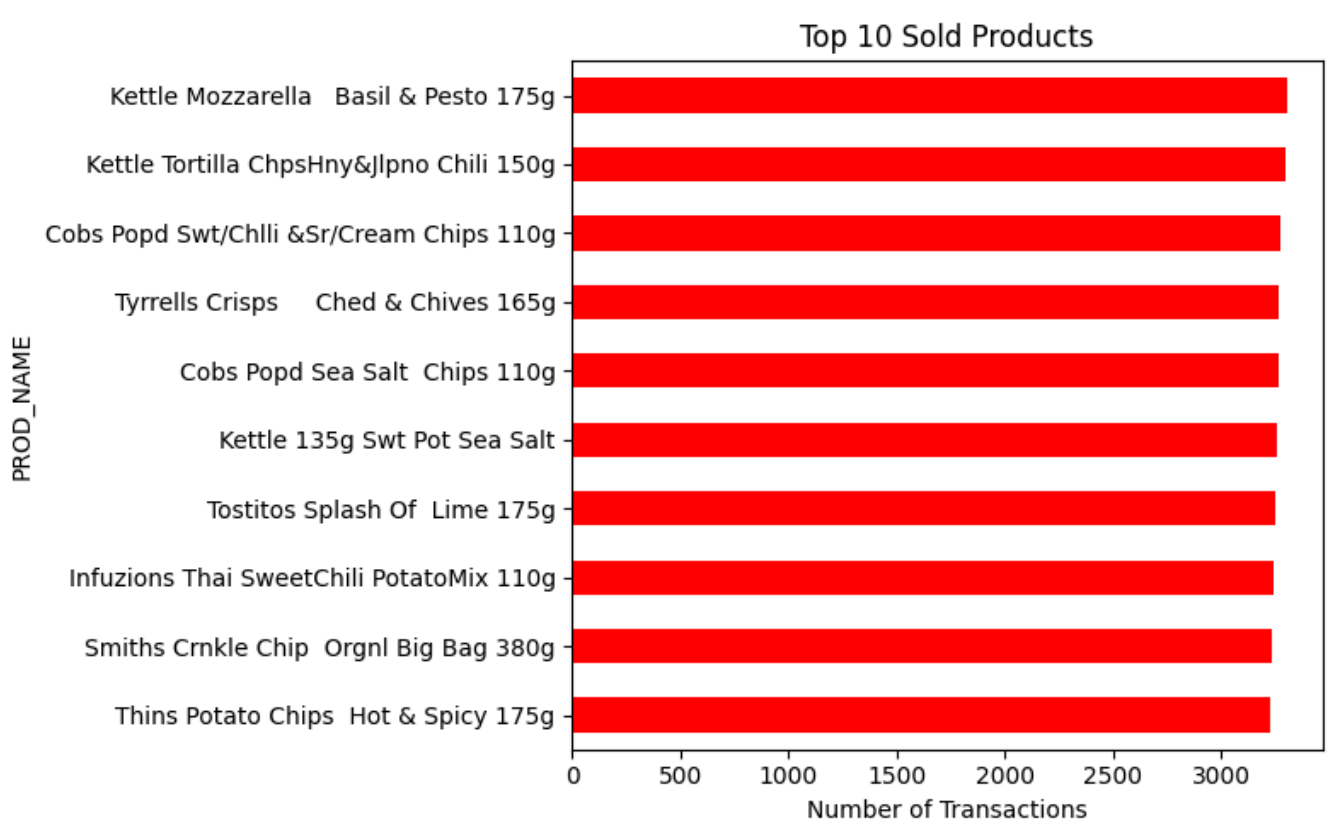
```
mp.tight_layout()
mp.show()
```




```
top_products = merged['PROD_NAME'].value_counts().head(10)
print("Top 10 Products by Number Transactions: ")
print(top_products)
```

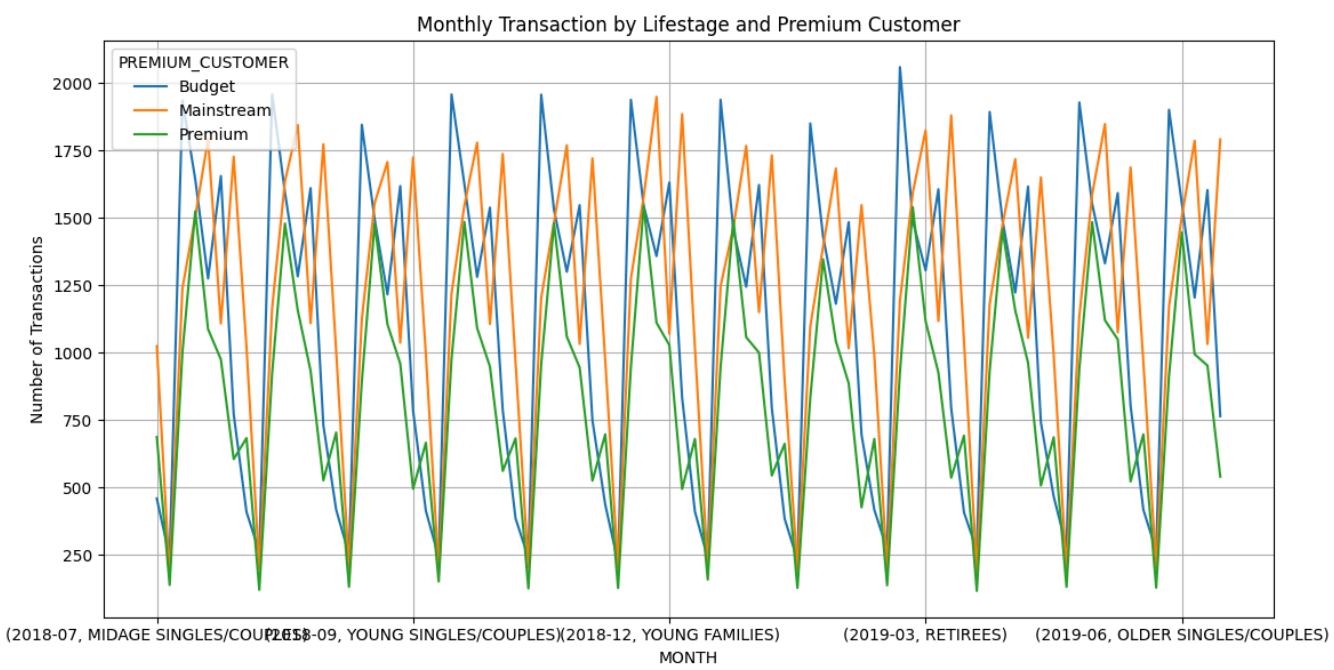
```
Top 10 Products by Number Transactions:
PROD_NAME
Kettle Mozzarella   Basil & Pesto 175g      3304
Kettle Tortilla ChpsHny&Jlpno Chili 150g    3296
Cobs Popd Swt/Chlli &Sr/Cream Chips 110g    3269
Tyrrells Crisps     Ched & Chives 165g      3268
Cobs Popd Sea Salt  Chips 110g              3265
Kettle 135g Swt Pot Sea Salt                3257
Tostitos Splash Of  Lime 175g               3252
Infuzions Thai SweetChili PotatoMix 110g    3242
Smiths Crnkle Chip  Orgnl Big Bag 380g      3233
Thins Potato Chips  Hot & Spicy 175g        3229
Name: count, dtype: int64
```

```
top_products.plot(kind='barh', title='Top 10 Sold Products',figsize=(8,5),color='red')
mp.xlabel('Number of Transactions')
mp.gca().invert_yaxis()
mp.tight_layout()
mp.show()
```




```
customer_stats = monthly_sales.groupby(['MONTH', 'LIFESTAGE', 'PREMIUM_CUSTOMER']).size().ur

customer_stats.plot(kind='line', figsize=(12,6), title='Monthly Transaction by Lifestage and
mp.xlabel('MONTH')
mp.ylabel('Number of Transactions')
mp.grid(True)
mp.tight_layout()
mp.show()
```



```
# 提取月份
merged['MONTH'] = merged['DATE'].dt.to_period('M')

# 每月销售总额和交易数量
monthly_sales = merged.groupby('MONTH').agg(
    total_sales=('TOT_SALES', 'sum'),
    transaction_count=('TXN_ID', 'count')
)

# 打印每月销售总额和交易数量
print(monthly_sales)
```

```
         total_sales  transaction_count
MONTH
2018-07    165275.30              22562
2018-08    158731.05              22411
2018-09    160522.00              21743
2018-10    164415.70              22288
2018-11    160233.70              21852
2018-12    167913.40              22835
2019-01    162642.30              22161
2019-02    150665.00              20405
2019-03    166265.20              22592
2019-04    159845.10              21766
2019-05    157367.65              22392
2019-06    160538.60              21829
```