```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Importing Libraries

```
import numpy as np
import random
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sn
import re
%matplotlib inline
```

Loading Data

```
df = pd.read_csv('/content/drive/MyDrive/Research/Datathon/DATATHON-UPDATED.txt', sep = '|')
```

Data Columns

```
df.columns
```

```
Index(['acct_id', 'VINTAGE_YEAR', 'STATE', 'GENERATION', 'AGE', 'FICO',
       'MONTHLY_INCOME', 'PRODUCT_SEGMENT', 'PARTNERBUSINESSKEY',
       'PARTNER_NAME', 'CREDIT_LIMIT', 'TRANSACTIONPOSTDT', 'TRANSACTIONAMT',
       'MERCHANTDESC', 'MCC', 'MCC_DESCRIPTION', 'SPENDCATEGORYNM', 'ENTRYNBR',
       'ME_BALANCE'],
      dtype='object')
```

Total Transactions

```
len(df)
```

```
3555741
```

Total Unique Customers

```
len(pd.unique(df['acct_id']))
```

```
50000
```

Missing values

```
df.isna().sum().sum()
```

```
7968
```

Missing value for each column

```
for i in df.columns:
  print(i,end=' : ')
  print(df[i].isna().sum())
```

```
acct_id : 0
VINTAGE_YEAR : 0
STATE : 1968
GENERATION : 0
AGE : 0
FICO : 0
MONTHLY_INCOME : 0
PRODUCT_SEGMENT : 0
PARTNERBUSINESSKEY : 0
PARTNER_NAME : 0
```

```
CREDIT_LIMIT : 0
TRANSACTIONPOSTDT : 0
TRANSACTIONAMT : 0
MERCHANTDESC : 1092
MCC : 0
MCC_DESCRIPTION : 4908
SPENDCATEGORYNM : 0
ENTRYNBR : 0
ME_BALANCE : 0
```

```
for x in range(1,50):
    print(x, end=' ')
    print(x-2-(2*(int((x-2)/7))),end='),(')
```

```
1 -1),(2 0),(3 1),(4 2),(5 3),(6 4),(7 5),(8 6),(9 5),(10 6),(11 7),(12 8),(13 9),(14 10),(15 11),(16 10),(17 11),(18 12),(19 13),(20 14
```

Feature Engineering

New Columns Date, Month, Days

```
df['date'] = pd.to_datetime(df['TRANSACTIONPOSTDT'])
month_day_count={1:0,2:31,3:59,4:90,5:120,6:151,7:181}
df['buy_month'] = df['date'].apply(lambda date:date.month)
df['time_series'] = df['date'].apply(lambda date:(date.day+month_day_count[date.month]))
df['buy_weekday'] = df['date'].apply(lambda date:date.day_name())
```

```
df['time_series'] = df['time_series'].apply(lambda x: x-2-(2*(int((x-2)/7))))
```

Merchant Name Extraction

```
df['Merchant_Name'] = df['MERCHANTDESC'].apply(lambda x: re.sub(r'[*|&|^|%|.|?|$|#|1|2|4|5|8|0]'," ",str(x)))
df['Merchant_Name'] = df['Merchant_Name'].apply(lambda x: str(x).strip().upper())
df['Merchant_Name'] = df['Merchant_Name'].apply(lambda x: x if (x[:3]=='THE' or x[:2]=='IN') else re.split(' ',str(x))[0] )
df['Merchant_Name'] = df['Merchant_Name'].apply(lambda x: 'AMAZON' if (x=='AMZN'  ) else x)
df['Merchant_Name'] = df['Merchant_Name'].apply(lambda x: 'WALMART' if (x=='WAL-MART' or x=='WM') else x)
df['Merchant_Name'] = df['Merchant_Name'].apply(lambda x: 'SAMSCLUB' if (x=='SAMS') else x)
```

```
mil=df[df['GENERATION']=='4. Millennials']
Gen_Z=df[df['GENERATION']=='5. Gen Z']
Boomers=df[df['GENERATION']=='2. Boomers']
```

```
both=df[(df['GENERATION']=='5. Gen Z')|(df['GENERATION']=='4. Millennials')]
```

Spend Category Name

```
pd.unique(Gen_Z['AGE'])
```

```
array([26, 24, 25, 23, 22, 21, 20])
```

Length

```
len(pd.unique(df['SPENDCATEGORYNM']))
```

```
23
```

```
pd.unique(df['MCC_DESCRIPTION'])
```

```
       'OMNI HOTEL ', 'MIRAGE HOTEL AND CASINO ', 'EMIRATES AIRLINES ',
       'HYATT PLACE', 'SHANGRI-LA INTERNATIONAL', 'AVIANCA', 'ANA HOTELS',
       'ELEC RAZOR STORES/SALE/SERV', 'MOTOR HOME DEALERS',
       'SILVER LEGACY HOTEL AND CASINO', 'CONTRACTORS - CONCRETE',
       'LOEWS HOTEL', 'EXCALIBUR HOTEL AND CASINO', 'AUTO PAINT SHOPS',
       'TENT AND AWNING SHOPS', "HARRAH'S HOTELS AND CASINOS",
       "CAESAR'S HOTEL AND CASINO", 'AUSTRIAN AIR',
       'VIRGIN RIVER HOTEL AND CASINO',
       'AIRPORTS, AIRPORT TERMINALS/FLYING FIELDS',
       'MONTE CARLO HOTEL AND CASINO', 'MILLENNIUM HOTELS',
       'CHINA AIRLINES', 'EUROP CAR', 'SNOWMOBILE DEALERS',
       'MICROTEL INNS & SUITES', 'COSMOPOLITAN OF LAS VEGAS',
       'RIO SUITES', 'MAINSTAY SUITES', 'SN BRUSSELS AIRLINES',
       'WILDERNESS HOTEL & RESORT', 'INTRA-GOVERNMENT PURCHASES',
       'IBERIA', 'ARIA', 'AIR MAROC', 'MOBILE HOME DEALERS',
       'STENOGRAPHIC SERVICES', 'PREMIER TRAVEL INNS',
       'SECURITY BROKERS/DEALERS', 'SMUGGLERS NOTCH RESORT',
       'RADISSON BLU', 'RAILROADS-FREIGHT', 'AIR-INDIA',
       'ROYAL KONA RESORT', 'PAYLESS CAR RENTAL', 'PARIS LAS VEGAS HOTEL',
       'INTER-CONTINENTAL', 'GOLDEN NUGGET', 'CARIBBEAN AIRLINES',
       'CLUB MED', 'ROSEN HOTELS AND RESORTS', 'LUXOR HOTEL AND CASINO',
       'EMBASSY HOTELS', 'TREASURE ISLAND HOTEL & CASINO',
       'RADISSON HOTEL', 'FONTAINEBLEAU RESORTS',
       'ECONOMY INNS OF AMERICA', 'THUNDERBIRD/RED LION',
       'SCANDIC HOTELS', 'AIR FRANCE', "BALLY'S HOTEL AND CASINO",
       'AEROFLOT', 'BAHAMASAIR', 'VENETIAN RESORT HOTEL CASINO',
       'ANTIQUE REPRODUCTION STORES', 'WALDORF', 'LA QUINTA RESORT',
       'FURRIERS AND FUR SHOPS', 'EASYJET', 'WIZZ AIRLINES',
       'TROPICANA RESORT & CASINO', 'NEW YORK-NEW YORK HOTEL+CASINO',
       'GAYLORD PALMS', 'CROSSLAND', 'SWISSOTEL', 'PAKISTAN',
       'VIRGIN ATLANTIC', 'BEAU RIVAGE HOTEL AND CASINO',
       'OUTBOUND TELEMARKETING MERCHNT', 'DIRECT MKTG-TRAVEL RELATED ARR',
       'MERIDIEN', 'SOFITEL HOTELS', 'FAIRFIELD HOTEL',
       'STRATOSPHERE HOTEL AND CASINO', 'FOUR SEASONS', 'MIDEASTAIR',
       'QATAR AIRWAYS', 'EGYPTAIR', 'OSTEOPATHS', 'HOTELES EL PRESIDENTE',
       'PAL AIR', 'NOVOTEL', 'GULF AIR (BAHRAIN)', 'LAN AIR',
       'HILTON CONRAD', 'KOREAN AIR', 'HARVEY/BRISTOL HOTELS',
       'TACA INTERNATIONAL', 'TAP AIR', 'HILTON INTERNATIONAL',
       'SAUDIA AIR', 'SHILO INN', 'AFFILIATED AUTO RENTAL',
       'BUDGET HOST INNS', 'DIRECT SELL/DOOR-TO-DOOR', 'WYNN LAS VEGAS',
       'CANOPY HOTELS', 'ALITALIA', 'SANDMAN HOTELS', 'LXR',
       'PEPPERMILL HOTEL CASINO', 'PARK INNS INTERNATIONAL',
       'THE PALACE HOTEL', "SAM'S TOWN HOTEL AND CASINO",
       'BILTMORE HOTEL & SUITES', "SHONEY'S INN", 'ROYAL HOTELS', 'SAS',
       'HOTELES MELIA', 'GRAND SIERRA RESORT', 'ETIHADAIR',
       'WINDWARD ISLAND', 'AUTOMOBILE SUPPLY STORES', 'ST. REGIS HOTEL',
       'ALA MOANA HOTEL', 'CATHAYPACAIR', 'SWISS INTERNATIONAL AIRLINES',
       'OXFORD SUITES', "WHISKEY PETE'S HOTEL & CASINO", 'SANDMAN INN',
       'AZUL AIR', 'CARLTON HOTELS', 'CITY LODGE HOTELS', 'STEIGENBERGER',
       'CHATEAU ELAN WINERY AND RESORT', 'PULLMAN INTERNATIONAL HOTELS',
       'FREMONT HOTEL AND CASINO'], dtype=object)
```

```
len(pd.unique(df['MCC_DESCRIPTION']))
```

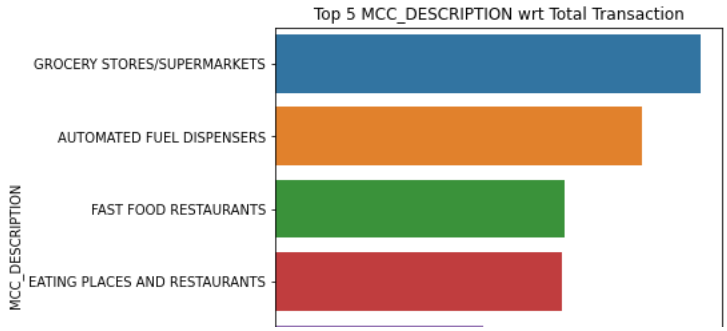
```
506
```

State

```
pd.unique(df['STATE'])
```

```
array(['AZ', 'NY', 'FL', 'TX', 'KY', 'MA', 'MI', 'CA', 'NE', 'SC', 'DC',
       'NC', 'NJ', 'VA', 'OK', 'NV', 'IA', 'MN', 'IL', 'SD', 'LA', 'NM',
       'MO', 'WA', 'MS', 'WV', 'MD', 'OH', 'TN', 'WI', 'IN', 'CO', 'GA',
       'AL', 'ID', 'HI', 'UT', 'KS', 'ND', 'PA', 'AR', 'MT', 'CT', 'NH',
       'OR', 'VT', 'ME', 'RI', 'DE', 'WY', nan, 'GU', 'PR', 'AK', 'AE',
       'TEX', 'VI', 'AP'], dtype=object)
```

```
x=df['MCC_DESCRIPTION'].value_counts()[:6].rename_axis('MCC_DESCRIPTION').reset_index(name='Total')
plt.figure(figsize = (6,6))
sn.barplot(x="Total", y="MCC_DESCRIPTION", data=x,label="MCC_DESCRIPTION").set(title='Top 5 MCC_DESCRIPTION wrt Total Transaction',xlabel="Am
plt.savefig("/content/drive/MyDrive/Research/Datathon/Figure/Top 5 MCC_DESCRIPTION wrt Total Transaction.png")
```




```
len(pd.unique(df['STATE']))
```

```
58
```

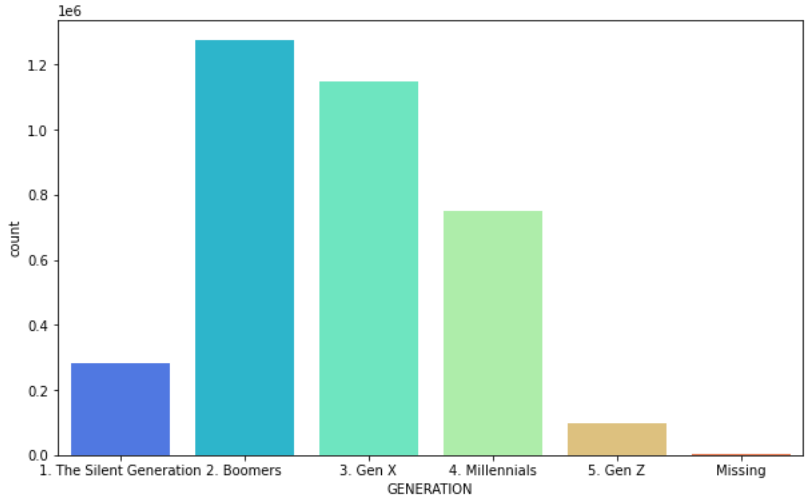
Generation

```
pd.unique(df['GENERATION'])
```

```
array(['1. The Silent Generation', '2. Boomers', '3. Gen X',
       '4. Millennials', '5. Gen Z', 'Missing'], dtype=object)
```

```
plt.figure(figsize = (10,6))
sn.countplot(x='GENERATION', data = df,  palette = 'rainbow')
```

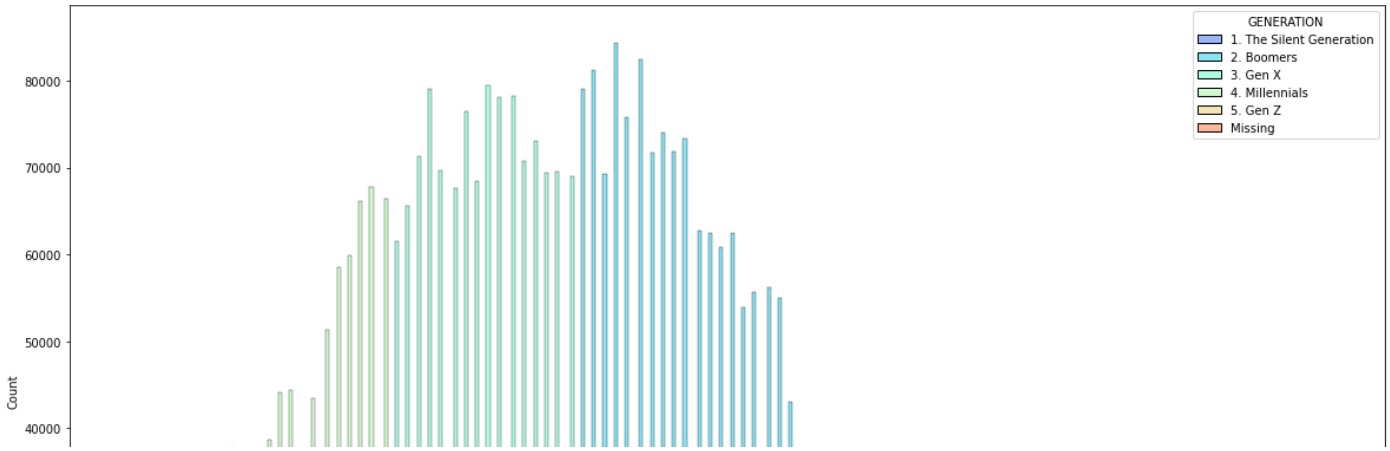
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1826ec4f50>
```



AGE

```
plt.figure(figsize = (20,12))
sn.histplot(x='AGE', data = df, hue='GENERATION', palette = 'rainbow')
```

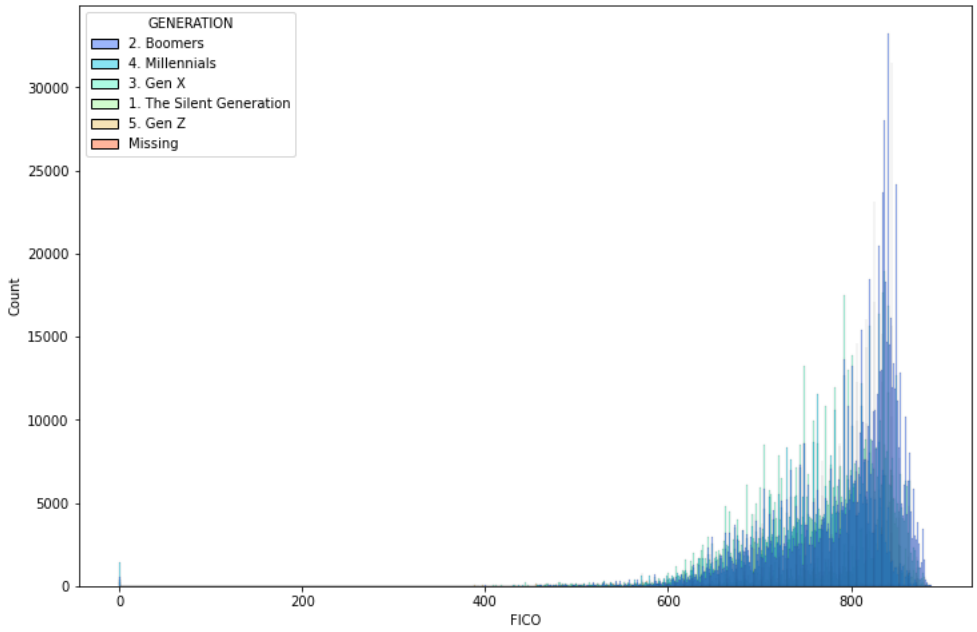
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbb3f8c1350>
```



FICO

```
plt.figure(figsize = (12,8))
sn.histplot(x='FICO', data = df, hue='GENERATION', palette = 'rainbow')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbfedf11490>
```



GEN_Z FICO

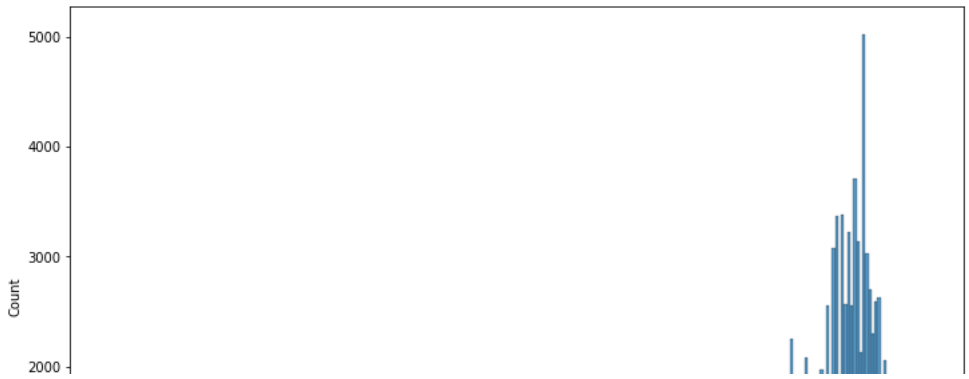
```
Gen_Z=df[df['GENERATION']=='5. Gen Z']
```

```
len(Gen_Z)
```

```
99588
```

```
plt.figure(figsize = (12,8))
sn.histplot(x='FICO', data = Gen_Z,  palette = 'rainbow')
```

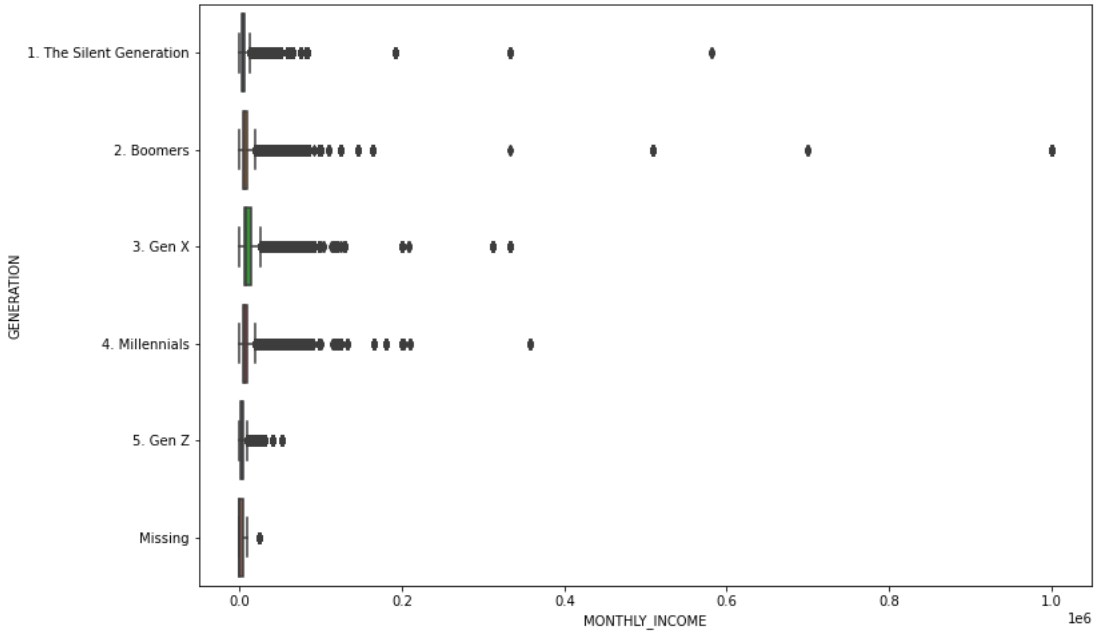
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbb3d953d10>
```



Monthly_Income

```
plt.figure(figsize = (12,8))
sn.boxplot(x='MONTHLY_INCOME', data = df, y='GENERATION')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbb3cfe8290>
```

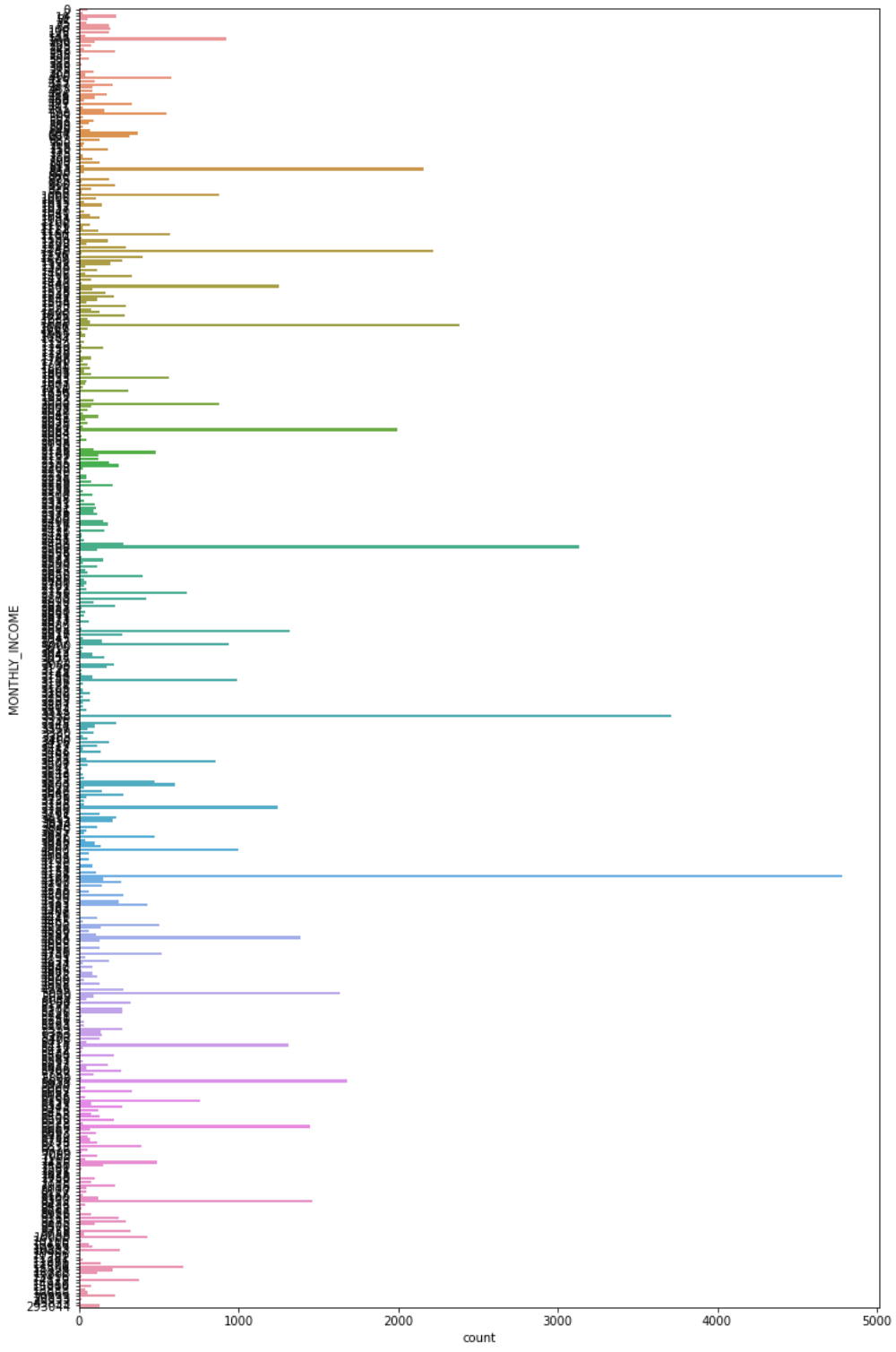



```
plt.figure(figsize = (12,8))
sn.boxplot(x='MONTHLY_INCOME', data = Gen_Z)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbb3d071890>
```

```
plt.figure(figsize = (12,20))
sn.countplot(y='MONTHLY_INCOME', data = Gen_Z)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbfa01eb0d0>
```



SPEND CATEGORY Name

```
pd.unique(df['SPENDCATEGORYNM'])
```

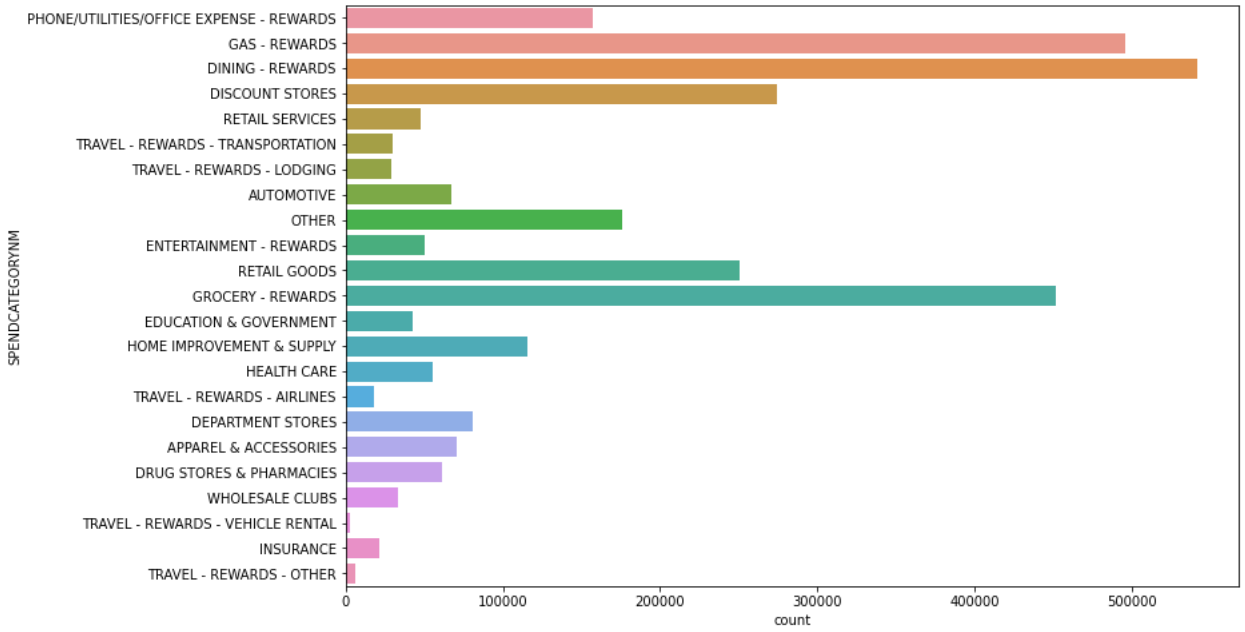
```
array(['PHONE/UTILITIES/OFFICE EXPENSE - REWARDS', 'GAS - REWARDS',
       'DINING - REWARDS', 'DISCOUNT STORES', 'RETAIL SERVICES',
       'TRAVEL - REWARDS - TRANSPORTATION', 'TRAVEL - REWARDS - LODGING',
```

```
'AUTOMOTIVE', 'OTHER', 'ENTERTAINMENT - REWARDS', 'RETAIL GOODS',
'GROCERY - REWARDS', 'EDUCATION & GOVERNMENT',
'HOME IMPROVEMENT & SUPPLY', 'HEALTH CARE',
'TRAVEL - REWARDS - AIRLINES', 'DEPARTMENT STORES',
'APPAREL & ACCESSORIES', 'DRUG STORES & PHARMACIES',
'WHOLESALE CLUBS', 'TRAVEL - REWARDS - VEHICLE RENTAL',
'INSURANCE', 'TRAVEL - REWARDS - OTHER'], dtype=object)
```

For ALL

```
plt.figure(figsize = (12,8))
sn.countplot(y='SPENDCATEGORYNM', data = df)
```

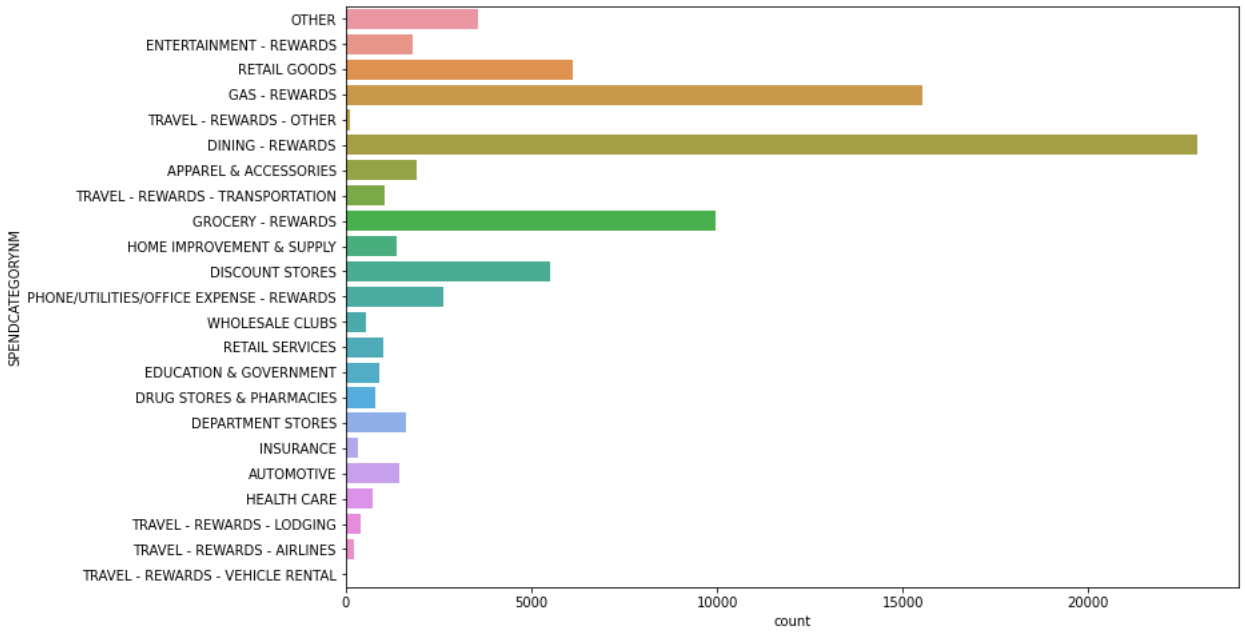
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbfa28c4c10>
```



Gen_z

```
plt.figure(figsize = (12,8))
sn.countplot(y='SPENDCATEGORYNM', data = Gen_Z)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbf94afbd50>
```



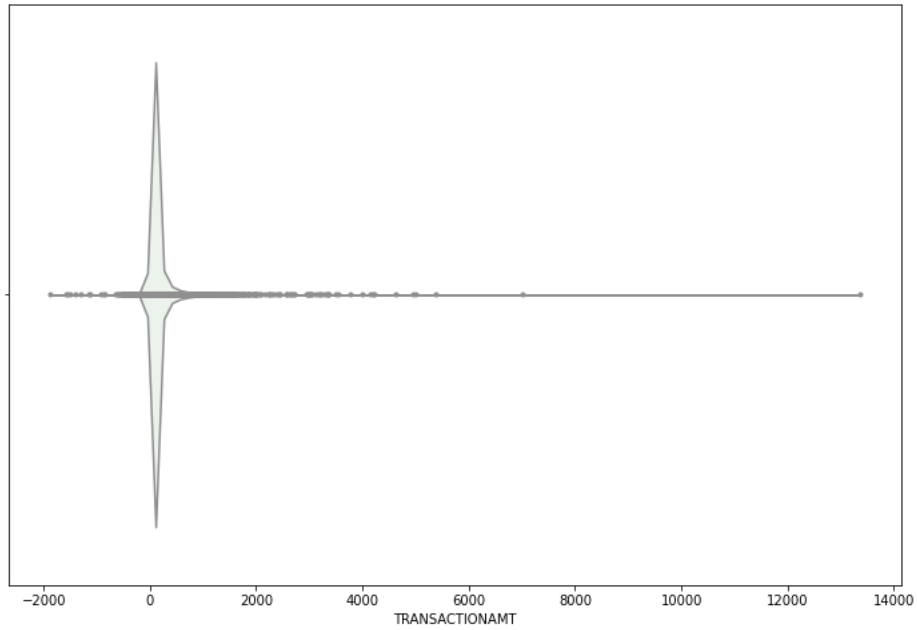
Spend Amount

```
Gen_Z['TRANSACTIONAMT'].describe()
```

```
count    80461.000000
mean        46.087254
std        138.685144
min      -1871.720000
25%          8.960000
50%         19.980000
75%         45.540000
max      13355.910000
Name: TRANSACTIONAMT, dtype: float64
```

```
plt.figure(figsize = (12,8))
sn.violinplot(data=Gen_Z,x='TRANSACTIONAMT', palette="light:g", inner="points", orient="h")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fab2282ccd0>
```



ME_BALANCE

```
Gen_Z['ME_BALANCE'].describe()
```

```
count    80461.000000
mean      1099.261547
std       1488.314599
min      -1375.940000
25%        219.430000
50%        604.500000
75%       1385.280000
max      13891.910000
Name: ME_BALANCE, dtype: float64
```

```
plt.figure(figsize = (12,8))
sn.violinplot(data=Gen_Z,x='ME_BALANCE', palette="light:g", inner="points", orient="h")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fab1c954710>
```

Gen_Z

```
Gen_Z=df[df['GENERATION']=='5. Gen Z']
```

'2. Boomers'

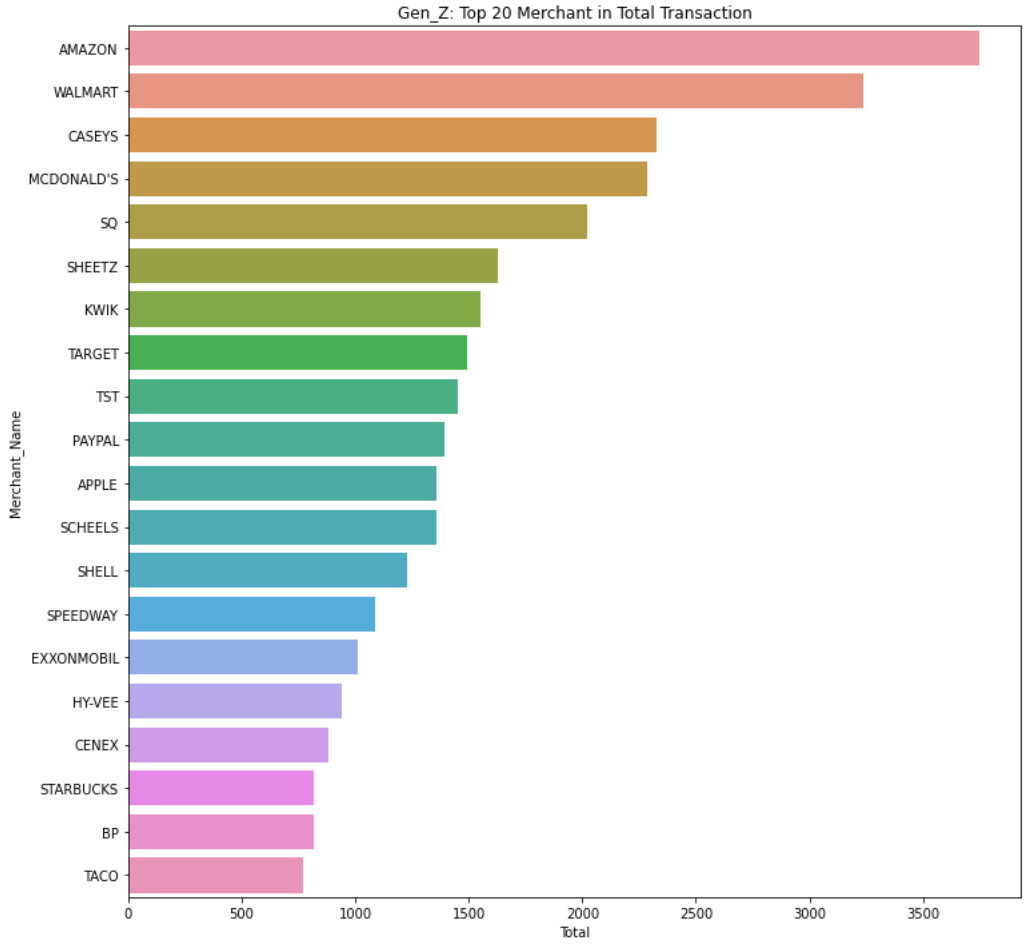
```
Boomers=df[df['GENERATION']=='2. Boomers']
```

```
x=Gen_Z['MCC_DESCRIPTION'].value_counts()[:20].rename_axis('MCC_DESCRIPTION').reset_index(name='Total')
plt.figure(figsize = (12,12))
sn.barplot(x="Total", y="MCC_DESCRIPTION", data=x,label="MCC_DESCRIPTION").set(title='Gen_Z: Top 20 MCC_DESCRIPTION wrt Total Transaction')
plt.savefig("/content/drive/MyDrive/Research/Datathon/Figure/Gen_Z Top 20 Top 20 MCC_DESCRIPTION wrt Total Transaction.png")
```

Top 20 Merchant wrt to Total transaction by Gen_Z

```
x=Gen_Z['Merchant_Name'].value_counts()[:20].rename_axis('Merchant_Name').reset_index(name='Total')
plt.figure(figsize = (12,12))
sn.barplot(x="Total", y="Merchant_Name", data=x,label="Merchant_Name").set(title='Gen_Z: Top 20 Merchant in Total Transaction')
#plt.savefig("/content/drive/MyDrive/Research/Datathon/Figure/Gen_Z Top 20 Merchant in Total Spent.png")
```

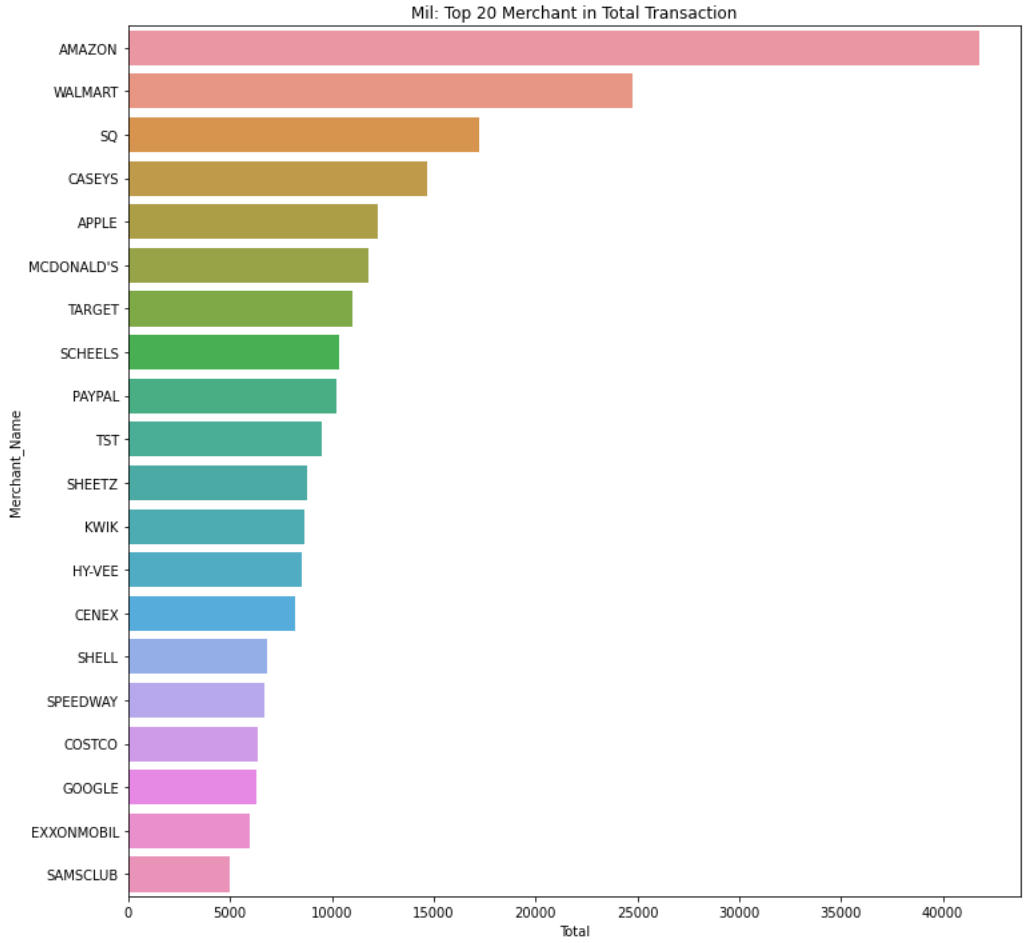
```
[Text(0.5, 1.0, 'Gen_Z: Top 20 Merchant in Total Transaction')]
```



MIL

```
x=mil['Merchant_Name'].value_counts()[:20].rename_axis('Merchant_Name').reset_index(name='Total')
plt.figure(figsize = (12,12))
sn.barplot(x="Total", y="Merchant_Name", data=x,label="Merchant_Name").set(title='Mil: Top 20 Merchant in Total Transaction')
#plt.savefig("/content/drive/MyDrive/Research/Datathon/Figure/Gen_Z Top 20 Merchant in Total Spent.png")
```

```
[Text(0.5, 1.0, 'Mil: Top 20 Merchant in Total Transaction')]
```




```
'SCHOOLS - DEFAULT''COMPUTERS/PERIPHERALS/SOFTWARE''RECREATION SERVICES''TOURIST ATTRACTIONS AND XHBT''STATIONERY STORES',
```

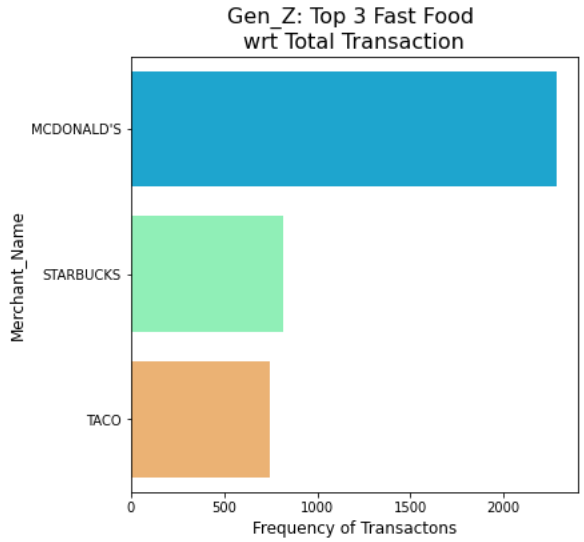
```
df['MCC_DESCRIPTION'].unique()
```

```
'INTER-CONTINENTAL ', 'GOLDEN NUGGET ', 'CARIBBEAN AIRLINES ',
'CLUB MED', 'ROSEN HOTELS AND RESORTS', 'LUXOR HOTEL AND CASINO',
'EMBASSY HOTELS', 'TREASURE ISLAND HOTEL & CASINO',
'RADISSON HOTEL', 'FONTAINEBLEAU RESORTS',
'ECONOMY INNS OF AMERICA', 'THUNDERBIRD/RED LION',
'SCANDIC HOTELS', 'AIR FRANCE', "BALLY'S HOTEL AND CASINO",
'AEROFLOT', 'BAHAMASAIR', 'VENETIAN RESORT HOTEL CASINO',
'ANTIQUE REPRODUCTION STORES', 'WALDORF', 'LA QUINTA RESORT',
'FURRIERS AND FUR SHOPS', 'EASYJET', 'WIZZ AIRLINES',
'TROPICANA RESORT & CASINO', 'NEW YORK-NEW YORK HOTEL+CASINO',
'GAYLORD PALMS', 'CROSSLAND', 'SWISSOTEL', 'PAKISTAN',
'VIRGIN ATLANTIC', 'BEAU RIVAGE HOTEL AND CASINO',
'OUTBOUND TELEMARKETING MERCHNT', 'DIRECT MKTG-TRAVEL RELATED ARR',
'MERIDIEN', 'SOFITEL HOTELS', 'FAIRFIELD HOTEL',
'STRATOSPHERE HOTEL AND CASINO', 'FOUR SEASONS', 'MIDEASTAIR',
'QATAR AIRWAYS', 'EGYPTAIR', 'OSTEOPATHS', 'HOTELES EL PRESIDENTE',
'PAL AIR', 'NOVOTEL', 'GULF AIR (BAHRAIN)', 'LAN AIR',
'HILTON CONRAD', 'KOREAN AIR', 'HARVEY/BRISTOL HOTELS',
'TACA INTERNATIONAL', 'TAP AIR', 'HILTON INTERNATIONAL',
'SAUDIA AIR', 'SHILO INN', 'AFFILIATED AUTO RENTAL',
'BUDGET HOST INNS', 'DIRECT SELL/DOOR-TO-DOOR', 'WYNN LAS VEGAS',
'CANOPY HOTELS', 'ALITALIA', 'SANDMAN HOTELS', 'LXR',
'PEPPERMILL HOTEL CASINO', 'PARK INNS INTERNATIONAL',
'THE PALACE HOTEL', "SAM'S TOWN HOTEL AND CASINO",
'BILTMORE HOTEL & SUITES', "SHONEY'S INN", 'ROYAL HOTELS', 'SAS',
'HOTELES MELIA', 'GRAND SIERRA RESORT', 'ETIHADAIR',
'WINDWARD ISLAND', 'AUTOMOBILE SUPPLY STORES', 'ST. REGIS HOTEL',
'ALA MOANA HOTEL', 'CATHAYPACAIR', 'SWISS INTERNATIONAL AIRLINES',
'OXFORD SUITES', "WHISKEY PETE'S HOTEL & CASINO", 'SANDMAN INN',
'AZUL AIR', 'CARLTON HOTELS', 'CITY LODGE HOTELS', 'STEIGENBERGER',
'CHATEAU ELAN WINERY AND RESORT', 'PULLMAN INTERNATIONAL HOTELS',
'FREMONT HOTEL AND CASINO'], dtype=object)
```

MCC_Description

```
x=data['Merchant_Name'].value_counts()[:3].rename_axis('Merchant_Name').reset_index(name='Total')
fig=plt.figure(figsize = (6,6))
plt.title('Gen_Z: Top 3 Fast Food \nwrt Total Transaction', fontsize=16)
plt.xlabel('xlabel', fontsize=12)
plt.ylabel('ylabel', fontsize=12)
sn.barplot(x="Total", y="Merchant_Name", data=x,label="Merchant_Name", palette='rainbow').set(ylabel="Merchant_Name",xlabel="Frequency of Trai
#plt.savefig("/content/drive/MyDrive/Research/Datathon/Figure/Gen_Z Top 20 Top 20 MCC_DESCRIPTION wrt Total Transaction.png")
```

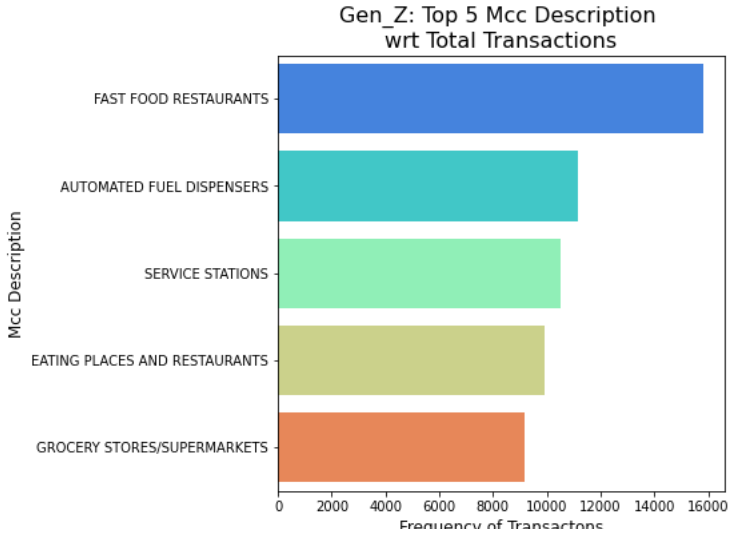
```
[Text(0, 0.5, 'Merchant_Name'), Text(0.5, 0, 'Frequency of Transactons')]
```



MIL

```
x=Gen_Z['MCC_DESCRIPTION'].value_counts()[:5].rename_axis('MCC_DESCRIPTION').reset_index(name='Total')
fig=plt.figure(figsize = (6,6))
plt.title('Gen_Z: Top 5 Mcc Description \nwrt Total Transactions', fontsize=16)
plt.xlabel('xlabel', fontsize=12)
plt.ylabel('ylabel', fontsize=12)
sn.barplot(x="Total", y="MCC_DESCRIPTION", data=x,label="Mcc Description", palette='rainbow').set(ylabel="Mcc Description",xlabel="Frequency
#plt.savefig("/content/drive/MyDrive/Research/Datathon/Figure/Gen_Z Top 20 Top 20 MCC_DESCRIPTION wrt Total Transaction.png")
```

```
[Text(0, 0.5, 'Mcc Description'), Text(0.5, 0, 'Frequency of Transactons')]
```

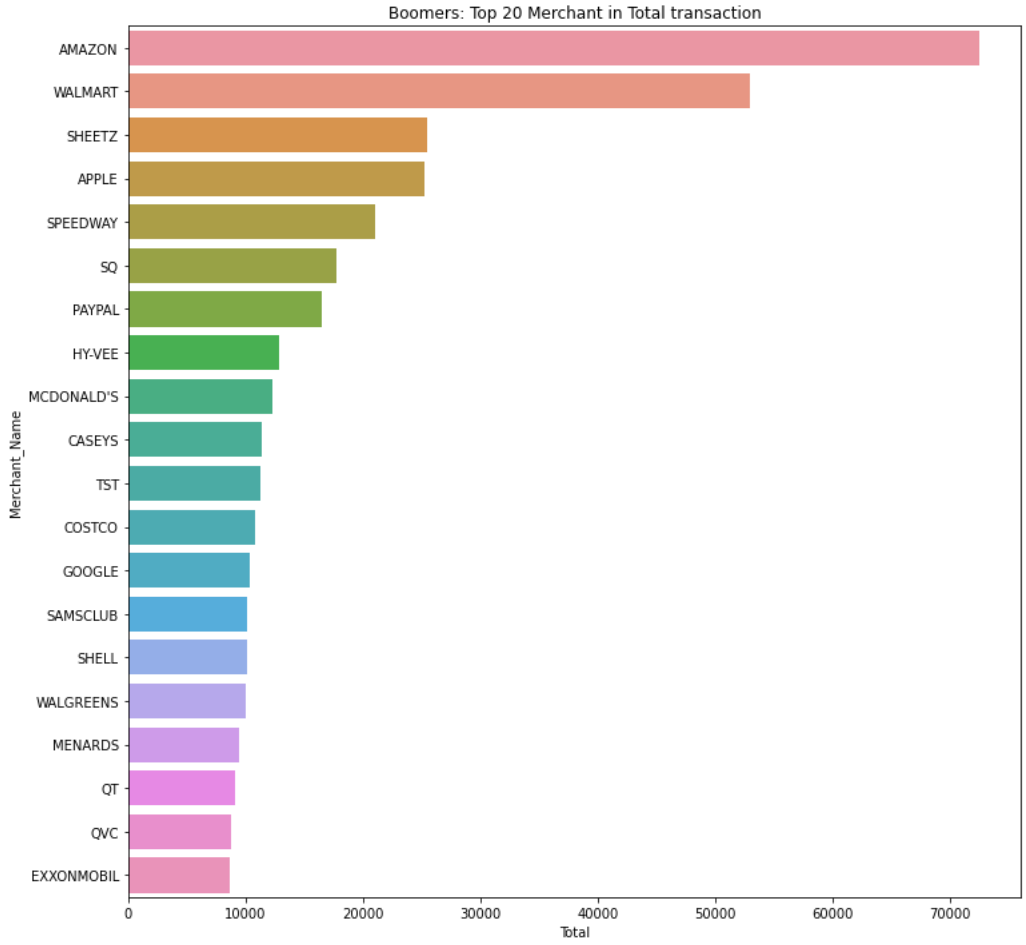



```
x['MCC_DESCRIPTION'].unique()
```

```
array(['FAST FOOD RESTAURANTS', 'AUTOMATED FUEL DISPENSERS',
       'SERVICE STATIONS', 'EATING PLACES AND RESTAURANTS',
       'GROCERY STORES/SUPERMARKETS'], dtype=object)
```

TOP 20 Merchant wrt total transaction for Boomers

```
x=Boomers['Merchant_Name'].value_counts()[:20].rename_axis('Merchant_Name').reset_index(name='Total')
plt.figure(figsize = (12,12))
sn.barplot(x="Total", y="Merchant_Name", data=x,label="Merchant_Name").set(title='Boomers: Top 20 Merchant in Total transaction')
plt.savefig("/content/drive/MyDrive/Research/Datathon/Figure/Boomers Top 20 Merchant in Total transaction.png")
```

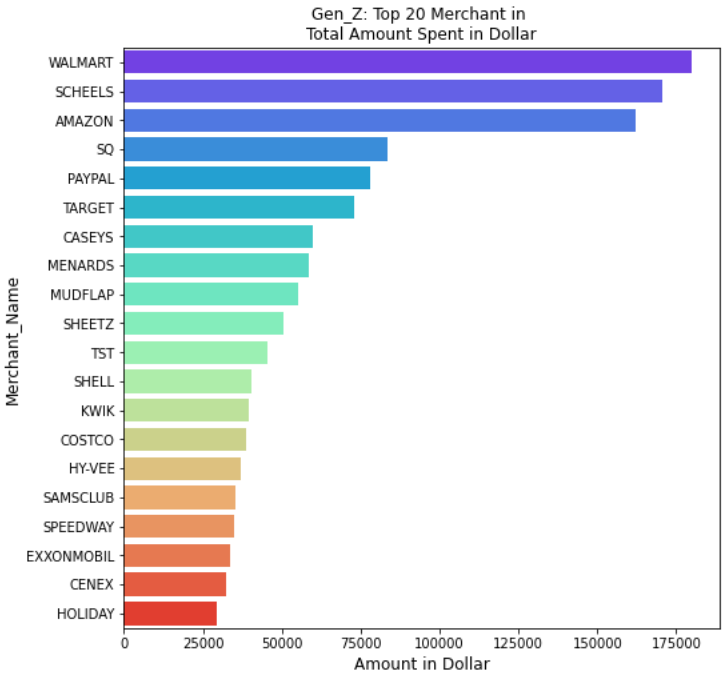


Top 20 Merchant wrt total spending Gen_Z

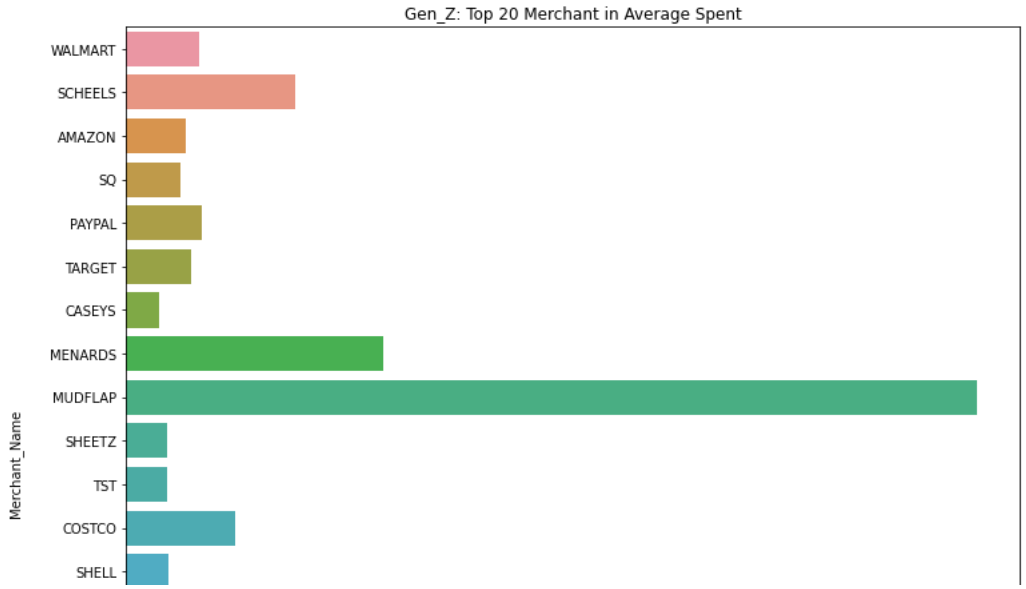
```
x=Gen_Z['Merchant_Name'].value_counts().rename_axis('Merchant_Name').reset_index(name='Total_Transactions')
avg_transaction = pd.DataFrame(Gen_Z.groupby('Merchant_Name')['TRANSACTIONAMT'].mean())
total_transaction = pd.DataFrame(Gen_Z.groupby('Merchant_Name')['TRANSACTIONAMT'].sum())
unique_customer = pd.DataFrame(Gen_Z.groupby('Merchant_Name')['acct_id'].nunique())
merchant_avg=pd.merge(x, avg_transaction, on = 'Merchant_Name')
merchant_all=pd.merge(merchant_avg, total_transaction, on = 'Merchant_Name')
merchant_all=pd.merge(merchant_all, unique_customer, on = 'Merchant_Name')
merchant_all_min_trans_30=merchant_all[merchant_all['Total_Transactions']>30]
merchant_all_min_trans_30=merchant_all_min_trans_30.rename(columns={"TRANSACTIONAMT_x": "AVG", "TRANSACTIONAMT_y": "SUM"})


Gen_Z_Top_merchant=merchant_all_min_trans_30.nlargest(n=20, columns=['SUM'])
```

```
fig=plt.figure(figsize = (8,8))
plt.title('Gen_Z: Top 5 Mcc Description wrt Total Transaction', fontsize=16)
plt.xlabel('xlabel', fontsize=12)
plt.ylabel('ylabel', fontsize=12)
sn.barplot(x="SUM", y="Merchant_Name", data=Gen_Z_Top_merchant,label="Merchant_Name", palette='rainbow').set(title='Gen_Z: Top 20 Merchant in

#plt.savefig("/content/drive/MyDrive/Research/Datathon/Figure/Gen_Z Top 20 Merchant in Total Spent.png")
```

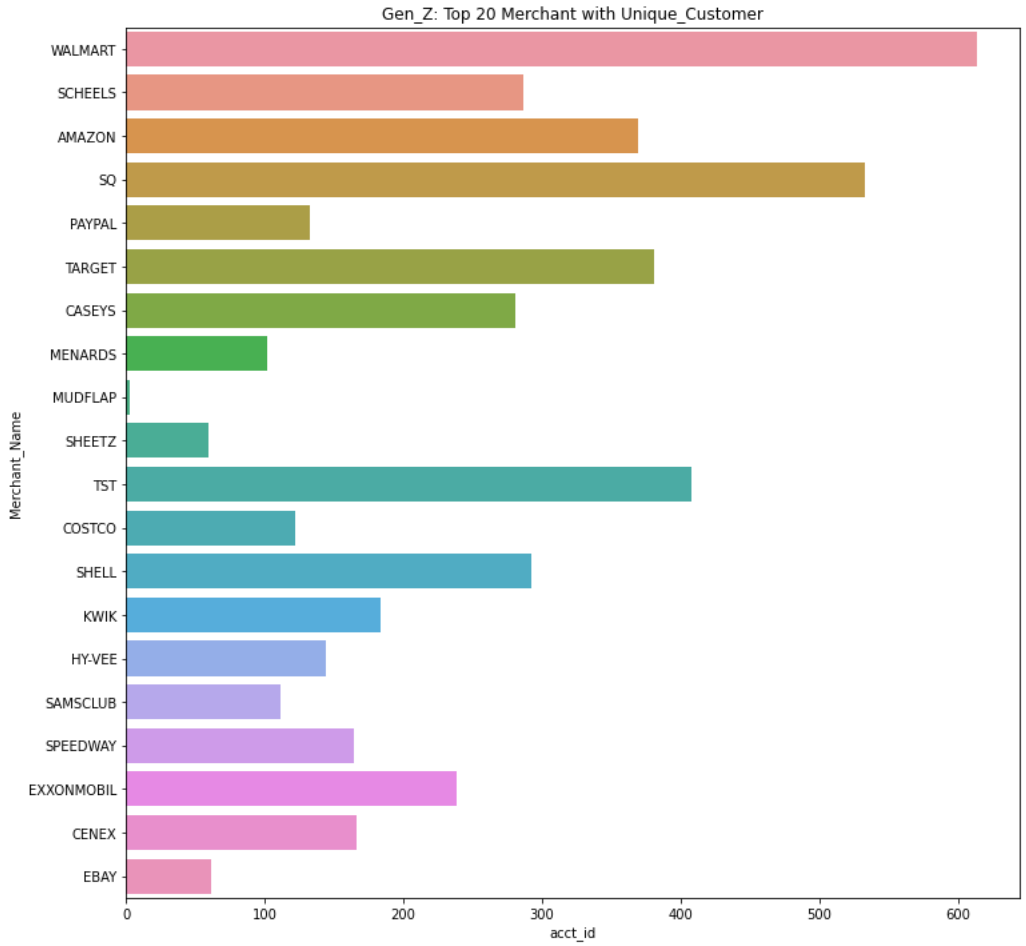
```
[Text(0.5, 0, 'Amount in Dollar'),
 Text(0.5, 1.0, 'Gen_Z: Top 20 Merchant in \nTotal Amount Spent in Dollar')]
```




```
plt.figure(figsize = (12,12))
sn.barplot(x="AVG", y="Merchant_Name", data=Gen_Z_Top_merchant,label="Merchant_Name").set(title='Gen_Z: Top 20 Merchant in Average Spent')

plt.savefig("/content/drive/MyDrive/Research/Datathon/Figure/Gen_Z Top 20 Merchant in Average Spent.png")
```




```
plt.figure(figsize = (12,12))
sn.barplot(x="acct_id", y="Merchant_Name", data=Gen_Z_Top_merchant,label="Merchant_Name").set(title='Gen_Z: Top 20 Merchant with Unique_Custor

plt.savefig("/content/drive/MyDrive/Research/Datathon/Figure/Gen_Z Top 20 Merchant with Unique_Customer.png")
```



MIL

```
x=Boomers['Merchant_Name'].value_counts().rename_axis('Merchant_Name').reset_index(name='Total_Transactions')
avg_transaction = pd.DataFrame(Boomers.groupby('Merchant_Name')['TRANSACTIONAMT'].mean())
total_transaction = pd.DataFrame(Boomers.groupby('Merchant_Name')['TRANSACTIONAMT'].sum())
unique_customer = pd.DataFrame(Boomers.groupby('Merchant_Name')['acct_id'].nunique())
merchant_avg=pd.merge(x, avg_transaction, on = 'Merchant_Name')
merchant_all=pd.merge(merchant_avg, total_transaction, on = 'Merchant_Name')
```

```
merchant_all=pd.merge(merchant_all, unique_customer, on = 'Merchant_Name')
merchant_all_min_trans_30=merchant_all[merchant_all['Total_Transactions']>30]
merchant_all_min_trans_30=merchant_all_min_trans_30.rename(columns={"TRANSACTIONAMT_x": "AVG", "TRANSACTIONAMT_y": "SUM"})
merchant_all_min_trans_30.head(4)
```

Double-click (or enter) to edit

```
Gen_Z.columns
```

```
Index(['acct_id', 'VINTAGE_YEAR', 'STATE', 'GENERATION', 'AGE', 'FICO',
       'MONTHLY_INCOME', 'PRODUCT_SEGMENT', 'PARTNERBUSINESSKEY',
       'PARTNER_NAME', 'CREDIT_LIMIT', 'TRANSACTIONPOSTDT', 'TRANSACTIONAMT',
       'MERCHANTDESC', 'MCC', 'MCC_DESCRIPTION', 'SPENDCATEGORYNM', 'ENTRYNBR',
       'ME_BALANCE', 'date', 'buy_month', 'time_series', 'buy_weekday',
       'Merchant_Name'],
      dtype='object')
```

```
Gen_Z[Gen_Z['Merchant_Name']=='APPLE']['SPENDCATEGORYNM'].unique()
```

```
array(['RETAIL GOODS', 'OTHER', 'AUTOMOTIVE', 'GROCERY - REWARDS',
       'TRAVEL - REWARDS - OTHER'], dtype=object)
```

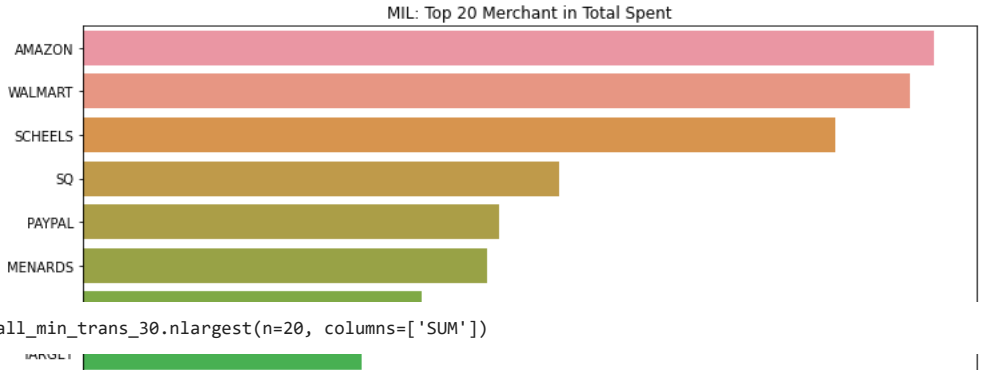
```
Gen_Z[Gen_Z['Merchant_Name']=='APPLE']['MCC_DESCRIPTION'].unique()
```

```
array(['LARGE DIGITAL GOODS MERCHANT', 'RECORD STORES',
       'DIGITAL GOODS BOOKSMOVIEMUSIC', 'ELECTRONIC SALES',
       'CIGAR STORES/STANDS', 'CAR & TRUCK DEALERS/NEW/USED',
       'GROCERY STORES/SUPERMARKETS', 'MISC FOOD STORES - DEFAULT',
       'TRAVEL AGENCIES'], dtype=object)
```

Top 20 Merchant by Boomers

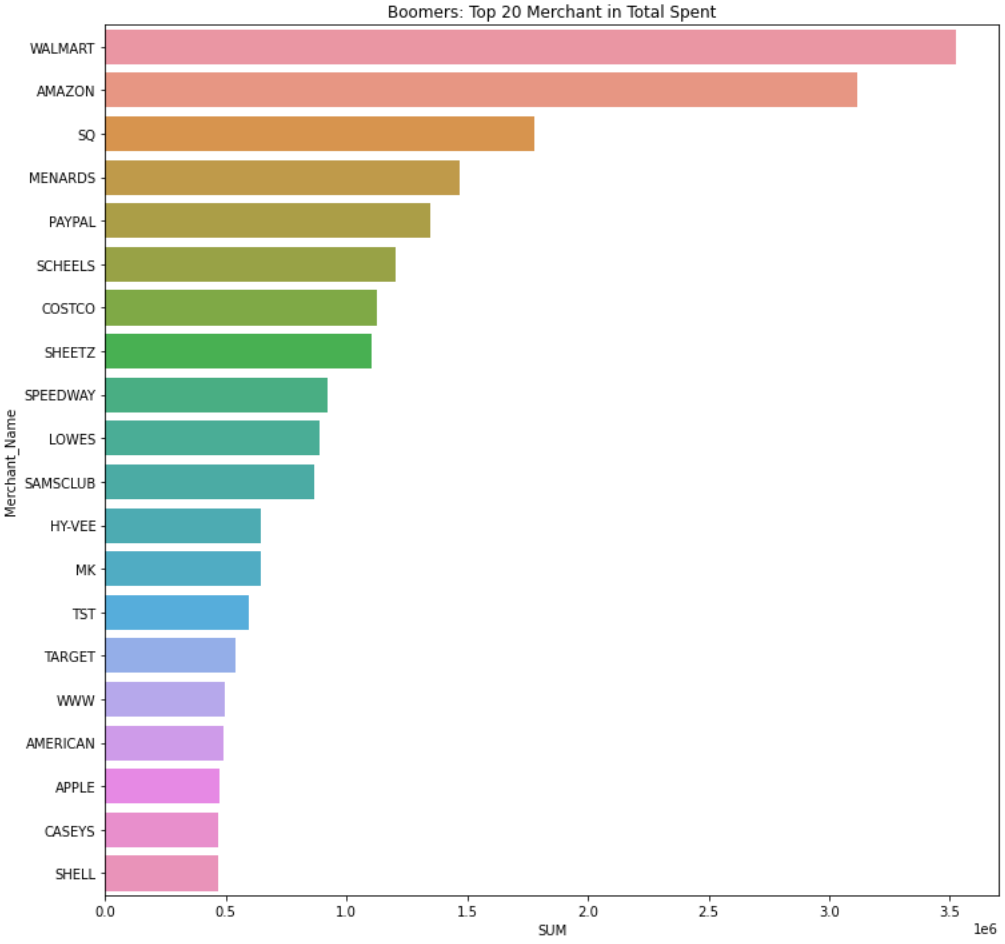
```
x=mil['Merchant_Name'].value_counts().rename_axis('Merchant_Name').reset_index(name='Total_Transactions')
avg_transaction = pd.DataFrame(mil.groupby('Merchant_Name')['TRANSACTIONAMT'].mean())
total_transaction = pd.DataFrame(mil.groupby('Merchant_Name')['TRANSACTIONAMT'].sum())
unique_customer = pd.DataFrame(mil.groupby('Merchant_Name')['acct_id'].nunique())
merchant_avg=pd.merge(x, avg_transaction, on = 'Merchant_Name')
merchant_all=pd.merge(merchant_avg, total_transaction, on = 'Merchant_Name')
merchant_all=pd.merge(merchant_all, unique_customer, on = 'Merchant_Name')
merchant_all_min_trans_30=merchant_all[merchant_all['Total_Transactions']>30]
merchant_all_min_trans_30=merchant_all_min_trans_30.rename(columns={"TRANSACTIONAMT_x": "AVG", "TRANSACTIONAMT_y": "SUM"})
merchant_all_min_trans_30.head(4)
Boomers_Top_merchant=merchant_all_min_trans_30.nlargest(n=20, columns=['SUM'])
plt.figure(figsize = (12,12))
sn.barplot(x="SUM", y="Merchant_Name", data=Boomers_Top_merchant,label="Merchant_Name").set(title='MIL: Top 20 Merchant in Total Spent')

#plt.savefig("/content/drive/MyDrive/Research/Datathon/Figure/Boomers Top 20 Merchant in Total Spent.png")
```

```
[Text(0.5, 1.0, 'MIL: Top 20 Merchant in Total Spent')]
```

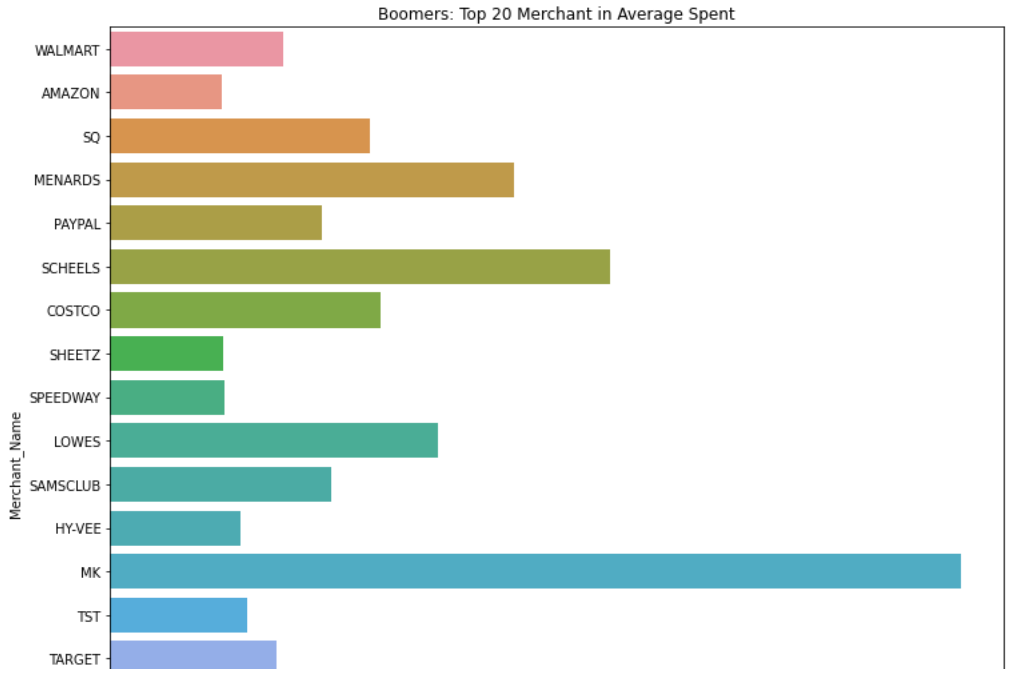



```
Boomers_Top_merchant=merchant_all_min_trans_30.nlargest(n=20, columns=['SUM'])
```

```
plt.figure(figsize = (12,12))
sn.barplot(x="SUM", y="Merchant_Name", data=Boomers_Top_merchant,label="Merchant_Name").set(title='Boomers: Top 20 Merchant in Total Spent')

plt.savefig("/content/drive/MyDrive/Research/Datathon/Figure/Boomers Top 20 Merchant in Total Spent.png")
```

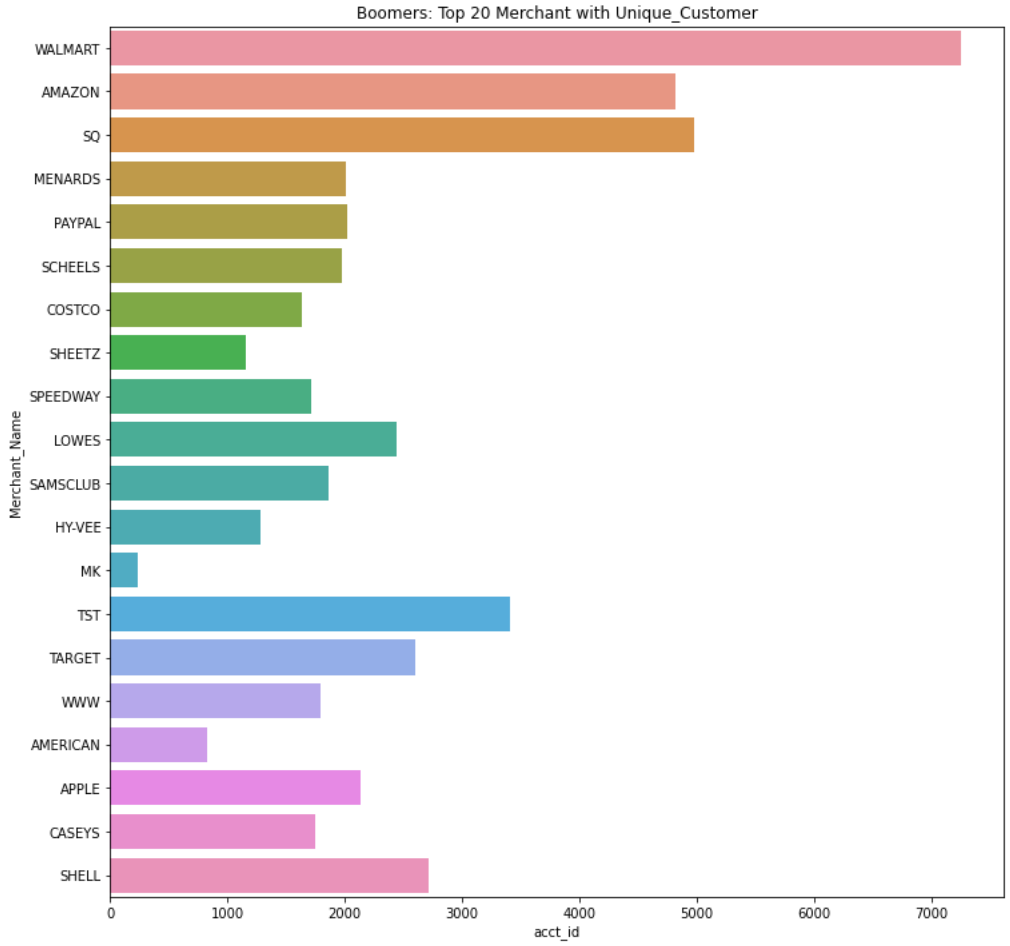



```
plt.figure(figsize = (12,12))
sn.barplot(x="AVG", y="Merchant_Name", data=Boomers_Top_merchant,label="Merchant_Name").set(title='Boomers: Top 20 Merchant in Average Spent'

plt.savefig("/content/drive/MyDrive/Research/Datathon/Figure/Boomers Top 20 Merchant in Total Spent.png")
```




```
plt.figure(figsize = (12,12))
sn.barplot(x="acct_id", y="Merchant_Name", data=Boomers_Top_merchant,label="Merchant_Name").set(title='Boomers: Top 20 Merchant with Unique_Cu
```

```
[Text(0.5, 1.0, 'Boomers: Top 20 Merchant with Unique_Customer')]
```

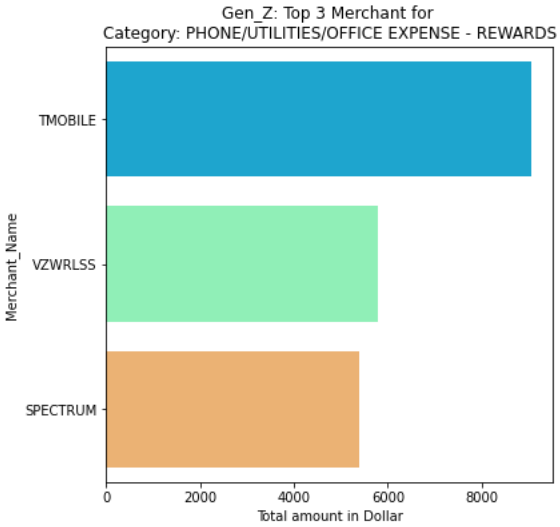


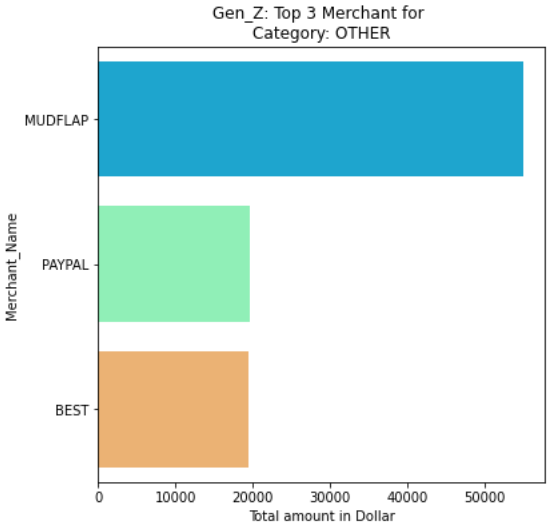
Top 5 merchant of each category on Total Spending

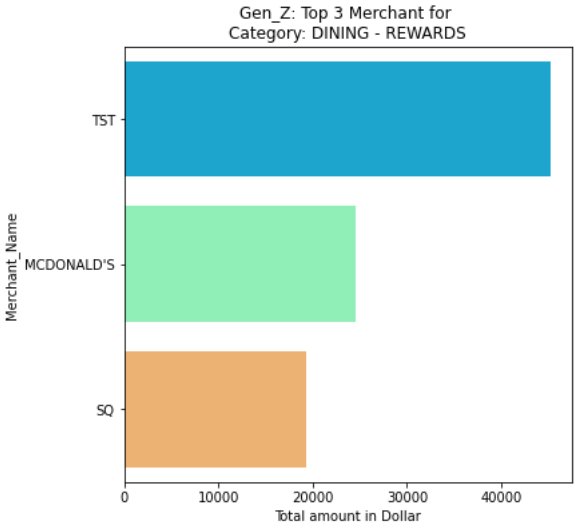
```
category=pd.unique(Gen_Z['SPENDCATEGORYNM'])
for item in category:
  data=Gen_Z[Gen_Z['SPENDCATEGORYNM']==item]
```

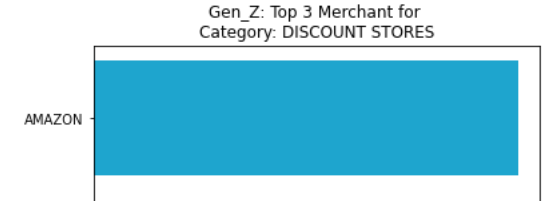
```
total_transaction = pd.DataFrame(data.groupby('Merchant_Name')['TRANSACTIONAMT'].sum())
x=total_transaction.nlargest(n=3, columns=['TRANSACTIONAMT']).reset_index('Merchant_Name')
plt.figure(figsize = (6,6))
titl='Gen_Z: Top 3 Merchant for \nCategory: '+item
des="/content/drive/MyDrive/Research/Datathon/Figure/Gen_Z "+item[:4]+" Top 3 Merchant for Category "+item[:4]+".png"
sn.barplot(x="TRANSACTIONAMT", y="Merchant_Name", data=x,label="Merchant", palette='rainbow').set(title=titl,xlabel='Total amount in Dollar
plt.savefig(des)

# data=Boomers[Boomers['SPENDCATEGORYNM']==item]
# total_transaction = pd.DataFrame(data.groupby('Merchant_Name')['TRANSACTIONAMT'].sum())
# x=total_transaction.nlargest(n=4, columns=['TRANSACTIONAMT']).reset_index('Merchant_Name')
# plt.figure(figsize = (6,6))
# titl='Boomers: Top 4 Merchant for Category: '+item
# des="/content/drive/MyDrive/Research/Datathon/Figure/Boomers "+item[:4]+" Top 4 Merchant for Category.png"
# sn.barplot(x="TRANSACTIONAMT", y="Merchant_Name", data=x,label="Merchant").set(title=titl)
# plt.savefig(des)

# data=mil[mil['SPENDCATEGORYNM']==item]
# total_transaction = pd.DataFrame(data.groupby('Merchant_Name')['TRANSACTIONAMT'].sum())
# x=total_transaction.nlargest(n=4, columns=['TRANSACTIONAMT']).reset_index('Merchant_Name')
# plt.figure(figsize = (6,6))
# titl='MIL: Top 4 Merchant for Category: '+item
# des="/content/drive/MyDrive/Research/Datathon/Figure/Boomers "+item[:4]+" Top 4 Merchant for Category.png"
# sn.barplot(x="TRANSACTIONAMT", y="Merchant_Name", data=x,label="Merchant").set(title=titl)
# #plt.savefig(des)
```

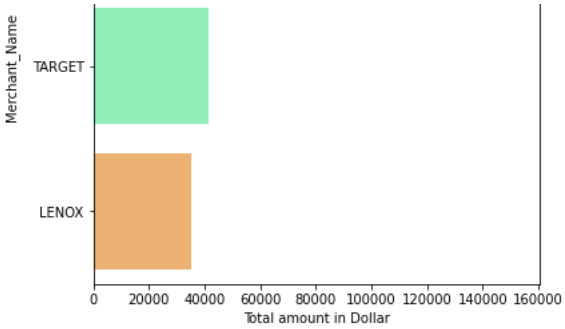
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: RuntimeWarning: More than 20 figures have been opened. Figures create

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: RuntimeWarning: More than 20 figures have been opened. Figures create

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: RuntimeWarning: More than 20 figures have been opened. Figures create
```



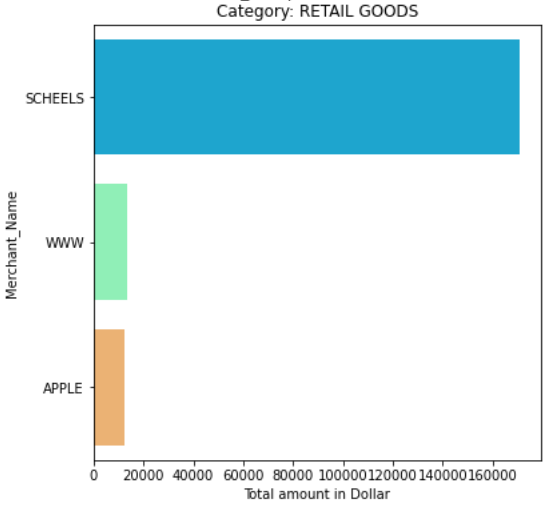




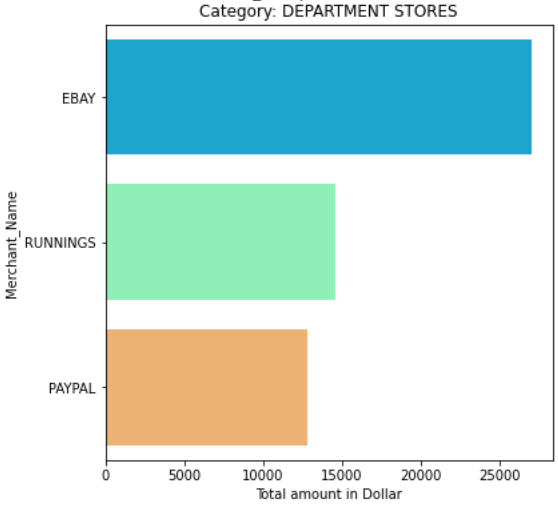




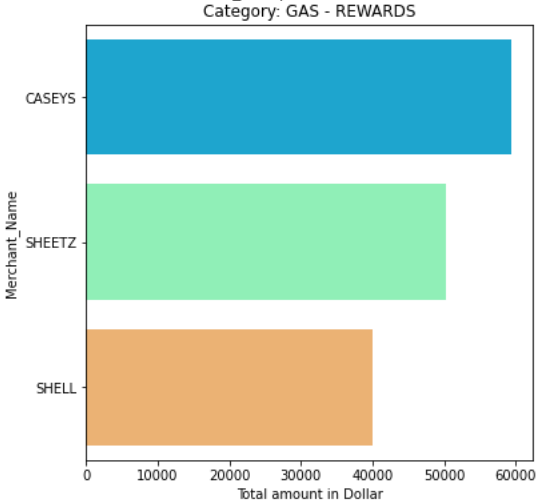
Gen_Z: Top 3 Merchant for
Category: RETAIL GOODS



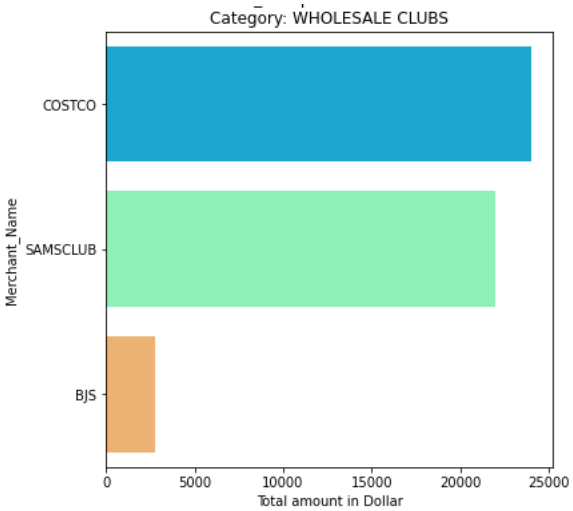
Gen_Z: Top 3 Merchant for
Category: DEPARTMENT STORES

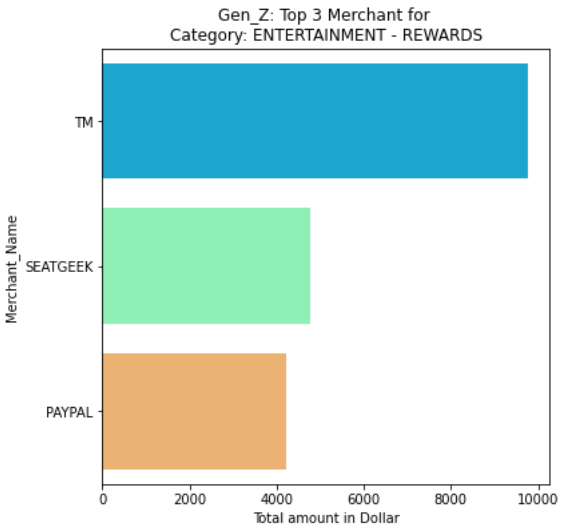


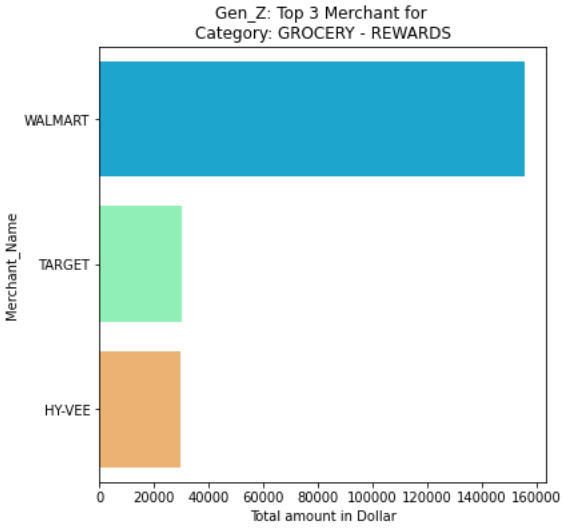
Gen_Z: Top 3 Merchant for
Category: GAS - REWARDS

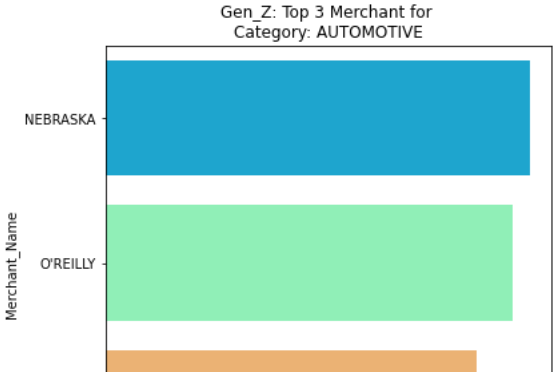


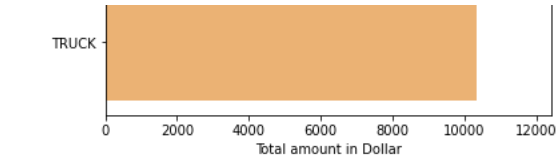
Gen_Z: Top 3 Merchant for

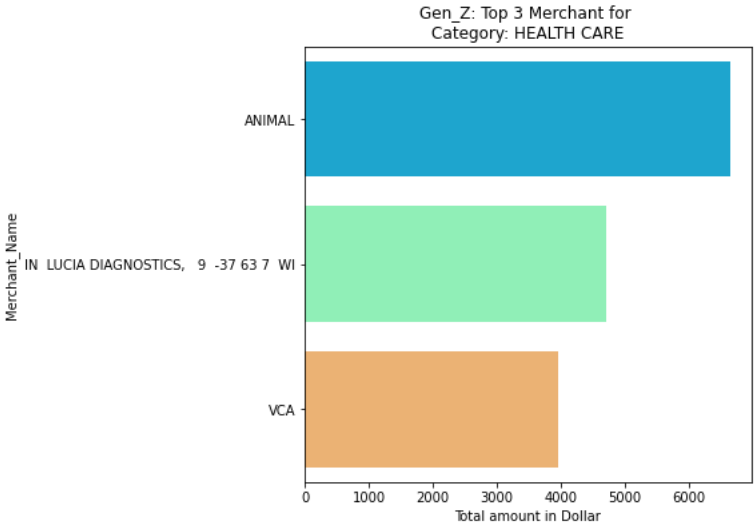

Category: WHOLESALE CLUBS
COSTCO
SAMSCLUB
BJS
Merchant_Name
0
5000
10000
15000
20000
25000
Total amount in Dollar

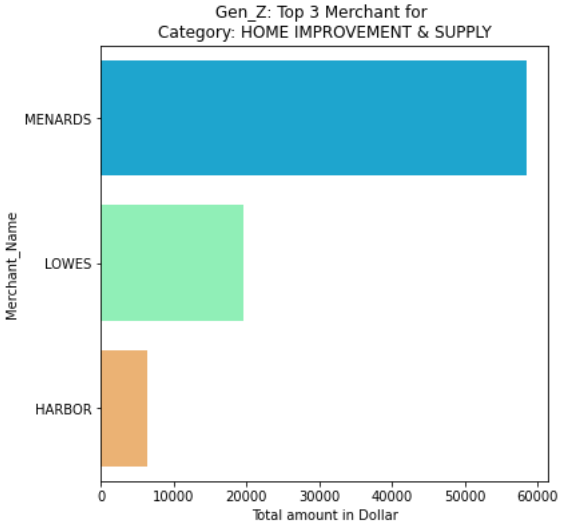

Gen_Z: Top 3 Merchant for
Category: ENTERTAINMENT - REWARDS
TM
SEATGEEK
PAYPAL
Merchant_Name
0
2000
4000
6000
8000
10000
Total amount in Dollar

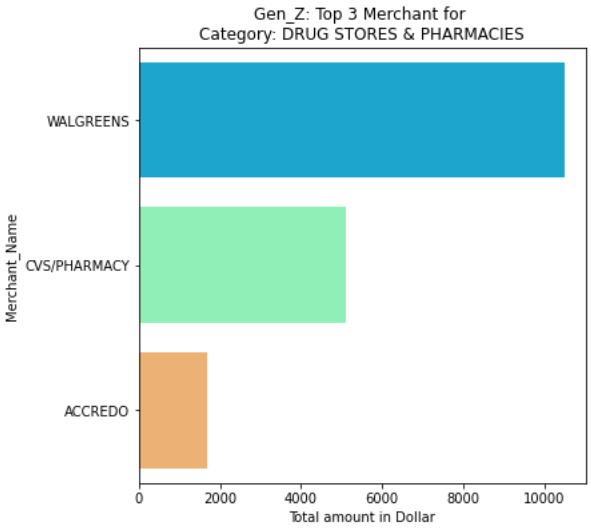

Gen_Z: Top 3 Merchant for
Category: GROCERY - REWARDS
WALMART
TARGET
HY-VEE
Merchant_Name
0
20000
40000
60000
80000
100000
120000
140000
160000
Total amount in Dollar

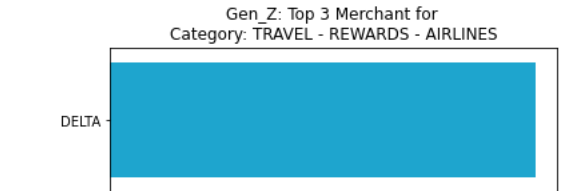

Gen_Z: Top 3 Merchant for
Category: AUTOMOTIVE
NEBRASKA
O'REILLY
Merchant_Name

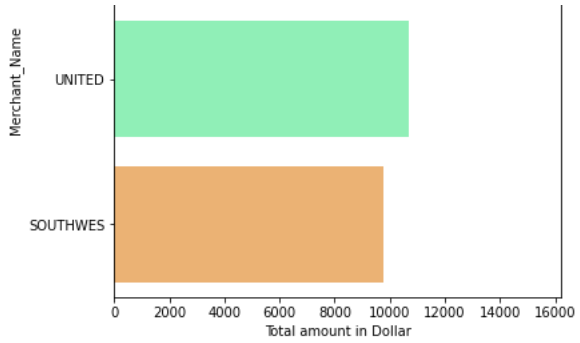

TRUCK
0
2000
4000
6000
8000
10000
12000
Total amount in Dollar

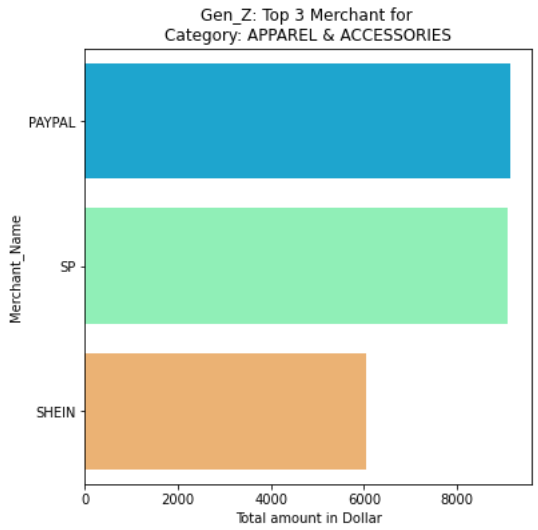

Gen_Z: Top 3 Merchant for
Category: HEALTH CARE
ANIMAL
IN LUCIA DIAGNOSTICS, 9 -37 63 7 WI
VCA
Merchant_Name
0
1000
2000
3000
4000
5000
6000
Total amount in Dollar

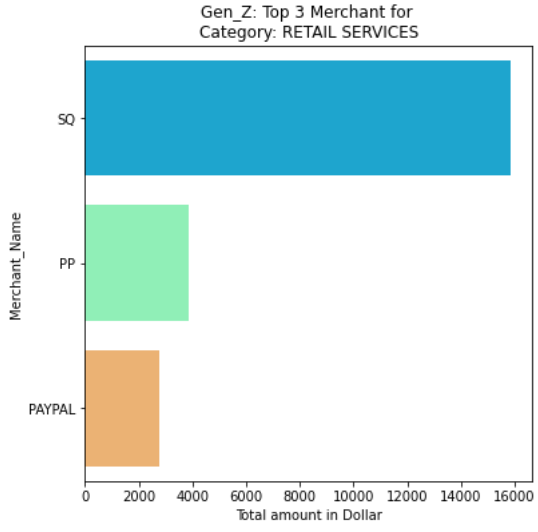

Gen_Z: Top 3 Merchant for
Category: HOME IMPROVEMENT & SUPPLY
MENARDS
LOWES
HARBOR
Merchant_Name
0
10000
20000
30000
40000
50000
60000
Total amount in Dollar

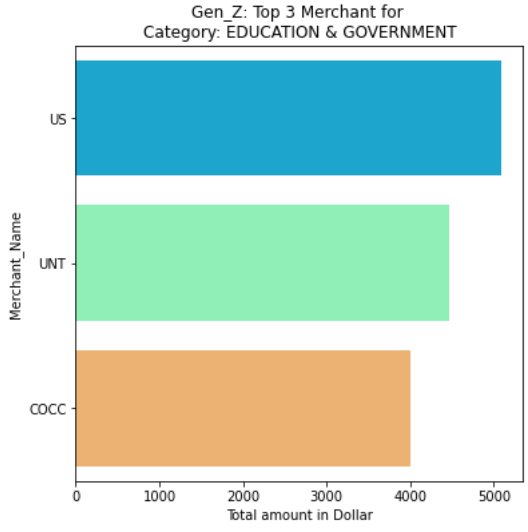

Gen_Z: Top 3 Merchant for
Category: DRUG STORES & PHARMACIES
WALGREENS
CVS/PHARMACY
ACCREDO
Merchant_Name
0
2000
4000
6000
8000
10000
Total amount in Dollar

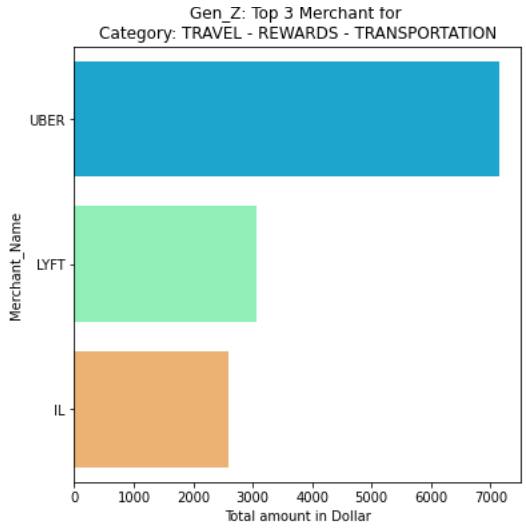

Gen_Z: Top 3 Merchant for
Category: TRAVEL - REWARDS - AIRLINES
DELTA

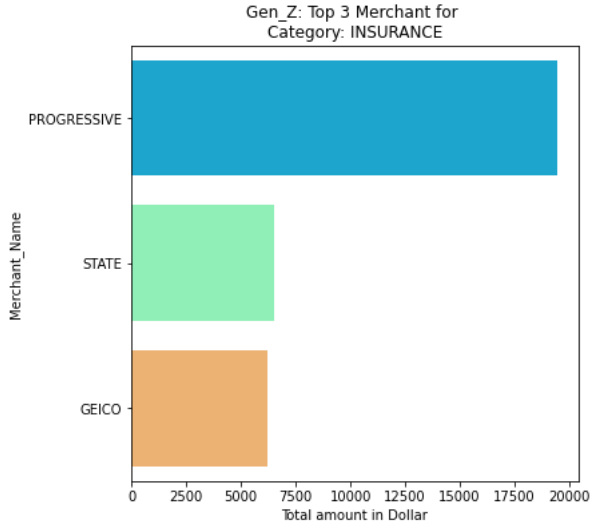

Merchant_Name
UNITED
SOUTHWES
0
2000
4000
6000
8000
10000
12000
14000
16000
Total amount in Dollar

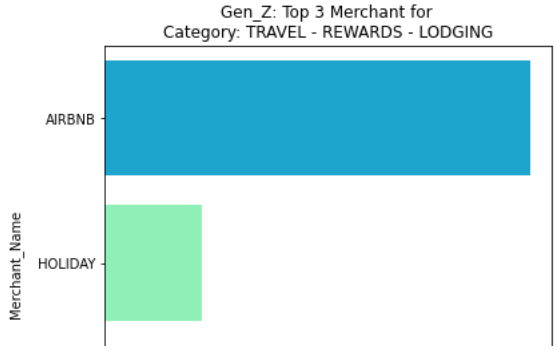

Gen_Z: Top 3 Merchant for
Category: APPAREL & ACCESSORIES
Merchant_Name
PAYPAL
SP
SHEIN
0
2000
4000
6000
8000
Total amount in Dollar


Gen_Z: Top 3 Merchant for
Category: RETAIL SERVICES
Merchant_Name
SQ
PP
PAYPAL
0
2000
4000
6000
8000
10000
12000
14000
16000
Total amount in Dollar


Gen_Z: Top 3 Merchant for
Category: EDUCATION & GOVERNMENT
Merchant_Name
US
UNT
COCC
0
1000
2000
3000
4000
5000
Total amount in Dollar





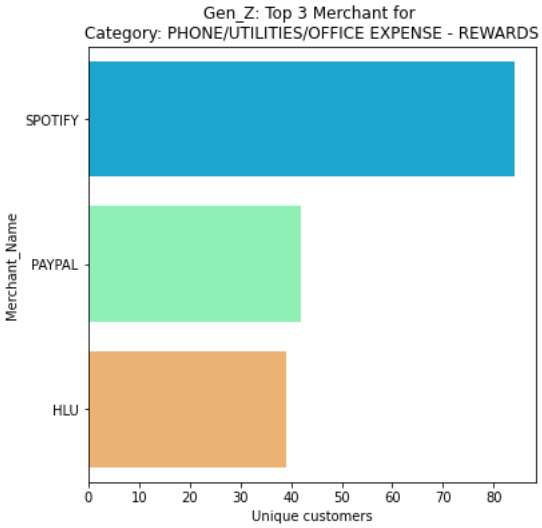


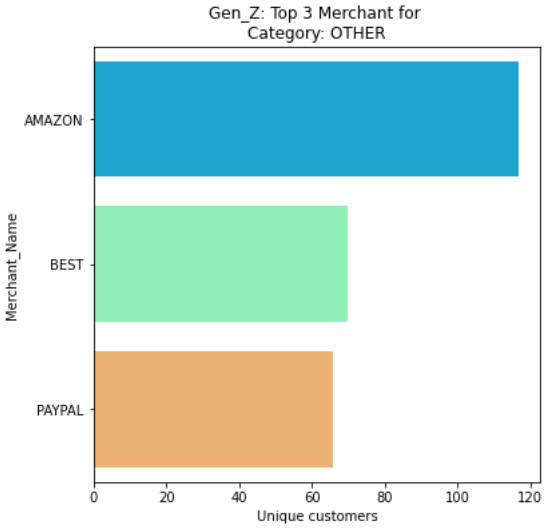
Top 4 Merchant on each category on Unique Customers:

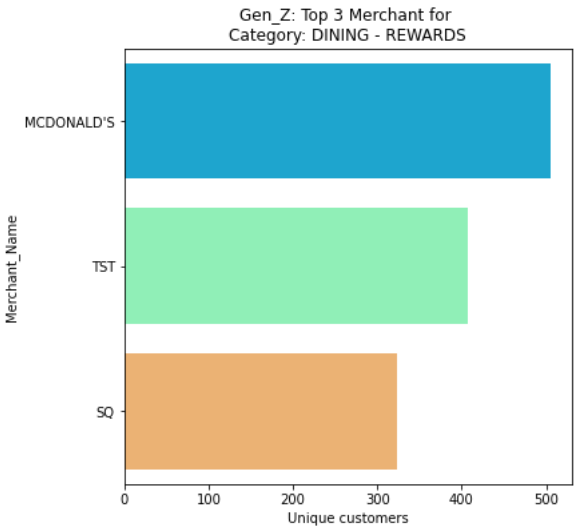

```
category=pd.unique(Gen_Z['SPENDCATEGORYNM'])
for item in category:
  data=Gen_Z[Gen_Z['SPENDCATEGORYNM']==item]
  unique_customer = pd.DataFrame(data.groupby('Merchant_Name')['acct_id'].nunique())
  x=unique_customer.nlargest(n=3, columns=['acct_id']).reset_index('Merchant_Name')
  plt.figure(figsize = (6,6))
  titl='Gen_Z: Top 3 Merchant for \nCategory: '+item
  des="/content/drive/MyDrive/Research/Datathon/Figure/Gen_Z "+item[:4]+" Top 3 Merchant on Unique Customers for Category "+item[:4]+".png"
  sn.barplot(x="acct_id", y="Merchant_Name", data=x,label="Merchant",palette='rainbow').set(title=titl,xlabel="Unique customers")
  plt.savefig(des)

  # data=Boomers[Boomers['SPENDCATEGORYNM']==item]
  # unique_customer = pd.DataFrame(data.groupby('Merchant_Name')['acct_id'].nunique())
  # x=unique_customer.nlargest(n=4, columns=['acct_id']).reset_index('Merchant_Name')
  # plt.figure(figsize = (6,6))
  # titl='Boomers: Top 4 Merchant for Category: '+item
  # des="/content/drive/MyDrive/Research/Datathon/Figure/Boomers "+item[:4]+" Top 4 on Unique Customers Merchant for Category.png"
  # sn.barplot(x="acct_id", y="Merchant_Name", data=x,label="Merchant").set(title=titl)
```

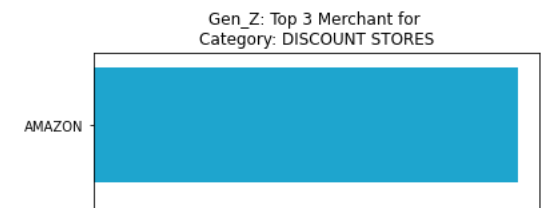
```
# plt.savefig(des)
# data=mil[mil['SPENDCATEGORYNM']==item]
# unique_customer = pd.DataFrame(data.groupby('Merchant_Name')['acct_id'].nunique())
# x=unique_customer.nlargest(n=4, columns=['acct_id']).reset_index('Merchant_Name')
# plt.figure(figsize = (6,6))
# titl='MIL: Top 4 Merchant for Category: '+item
# des="/content/drive/MyDrive/Research/Datathon/Figure/Boomers "+item[:4]+" Top 4 on Unique Customers Merchant for Category.png"
# sn.barplot(x="acct_id", y="Merchant_Name", data=x,label="Merchant").set(title=titl)
# #plt.savefig(des)
```

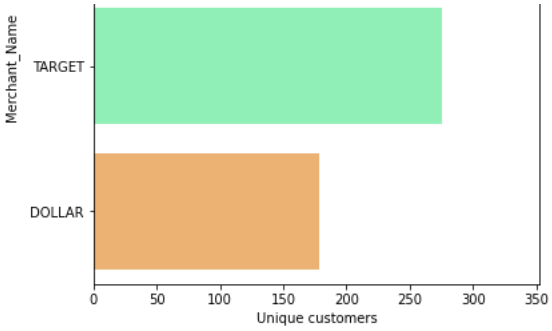
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: RuntimeWarning: More than 20 figures have been opened. Figures create
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: RuntimeWarning: More than 20 figures have been opened. Figures create
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: RuntimeWarning: More than 20 figures have been opened. Figures create
```



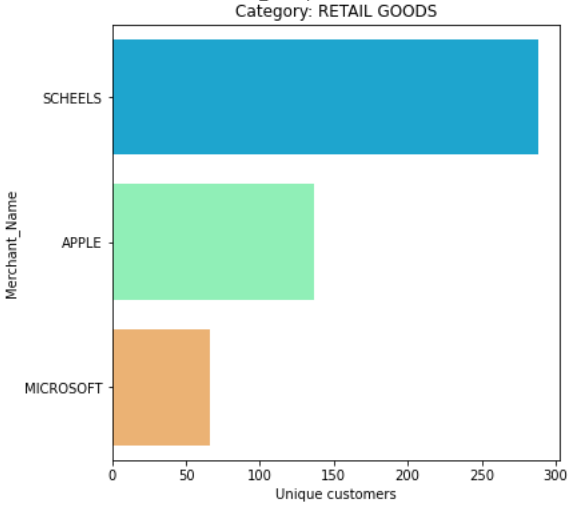




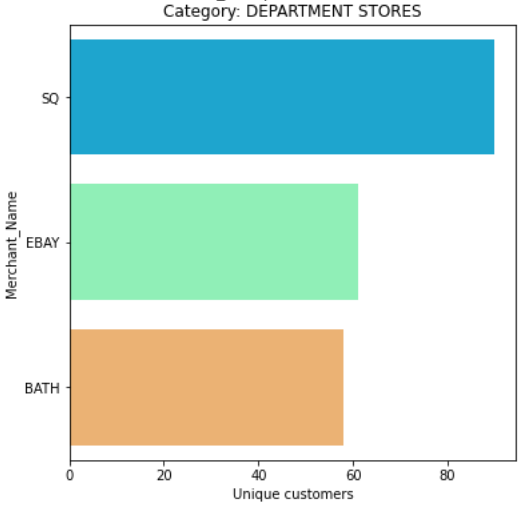




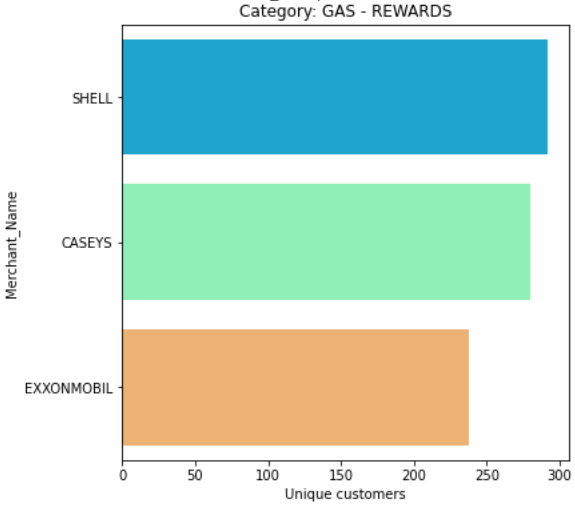
Gen_Z: Top 3 Merchant for
Category: RETAIL GOODS



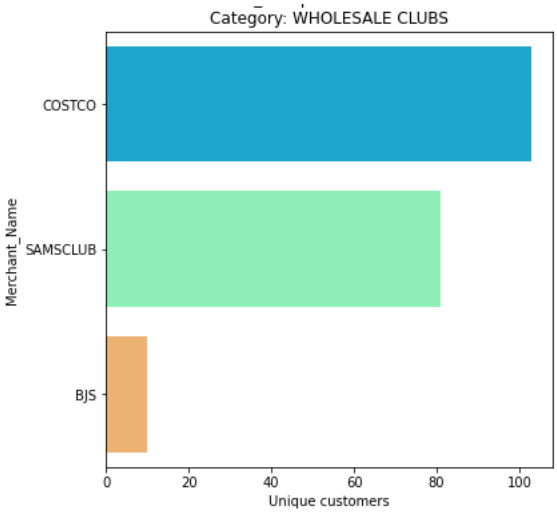
Gen_Z: Top 3 Merchant for
Category: DEPARTMENT STORES

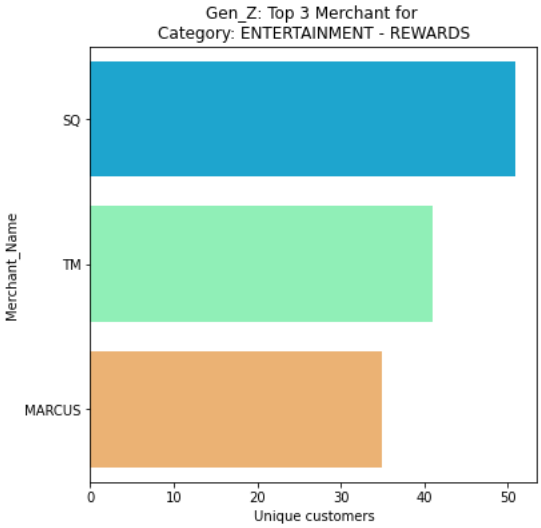


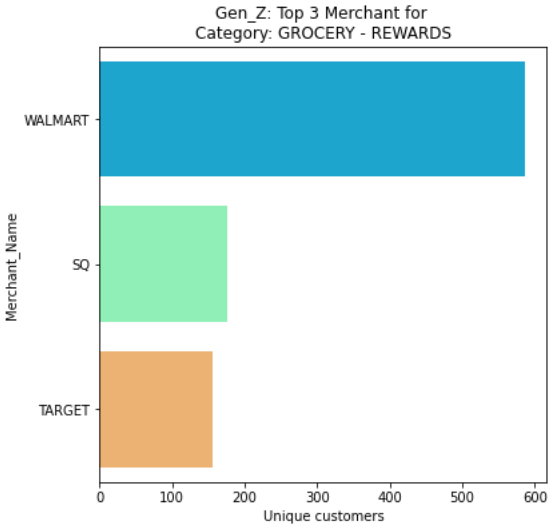
Gen_Z: Top 3 Merchant for
Category: GAS - REWARDS

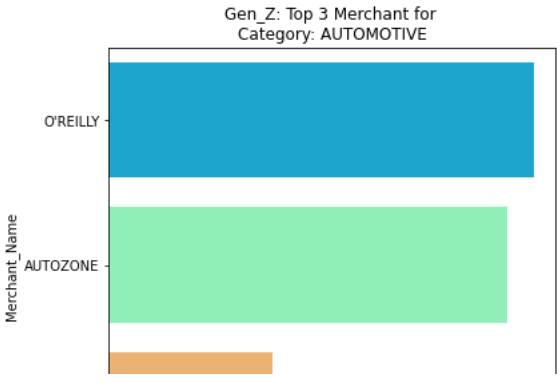


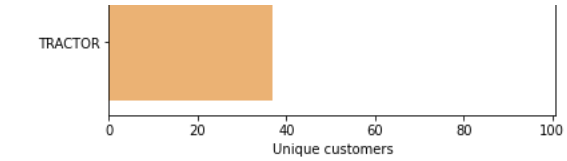
Gen_Z: Top 3 Merchant for

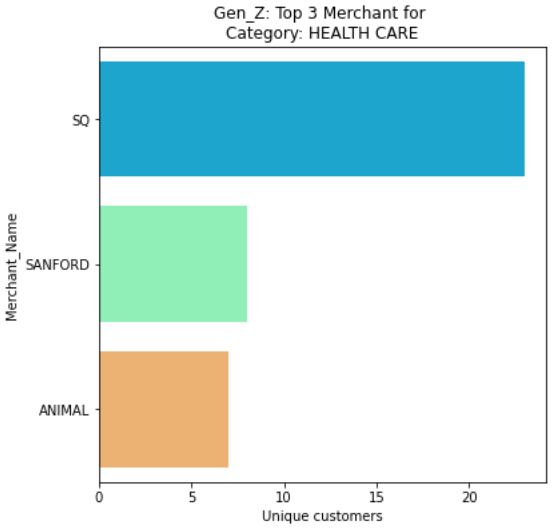

Category: WHOLESALE CLUBS
COSTCO
SAMSCLUB
BJS
Merchant_Name
0
20
40
60
80
100
Unique customers

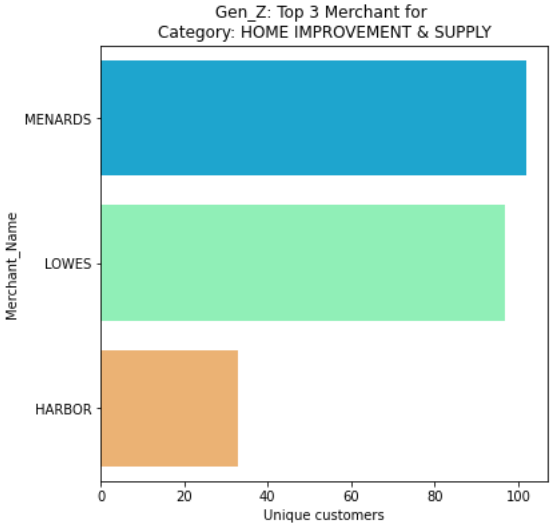

Gen_Z: Top 3 Merchant for
Category: ENTERTAINMENT - REWARDS
SQ
TM
MARCUS
Merchant_Name
0
10
20
30
40
50
Unique customers

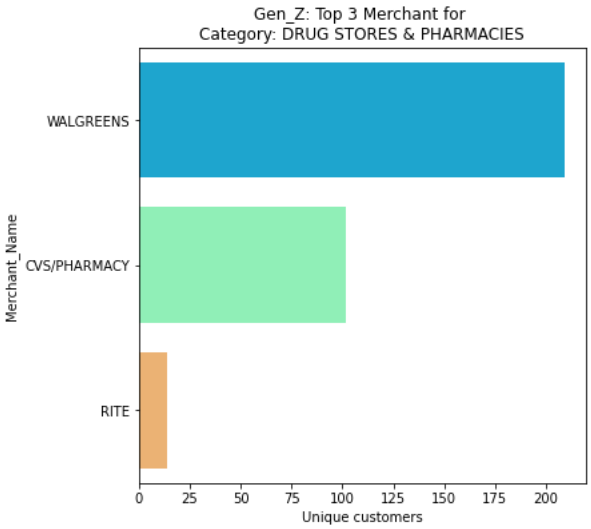

Gen_Z: Top 3 Merchant for
Category: GROCERY - REWARDS
WALMART
SQ
TARGET
Merchant_Name
0
100
200
300
400
500
600
Unique customers

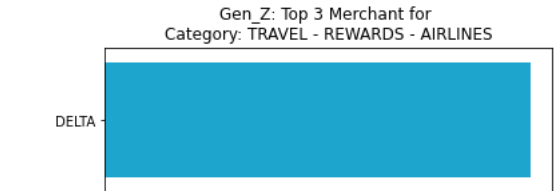

Gen_Z: Top 3 Merchant for
Category: AUTOMOTIVE
O'REILLY
AUTOZONE
Merchant_Name

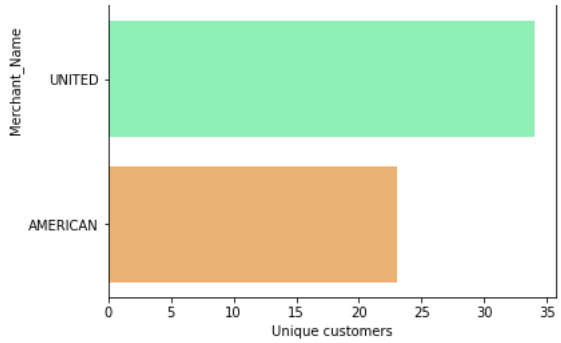

TRACTOR
0
20
40
60
80
100
Unique customers

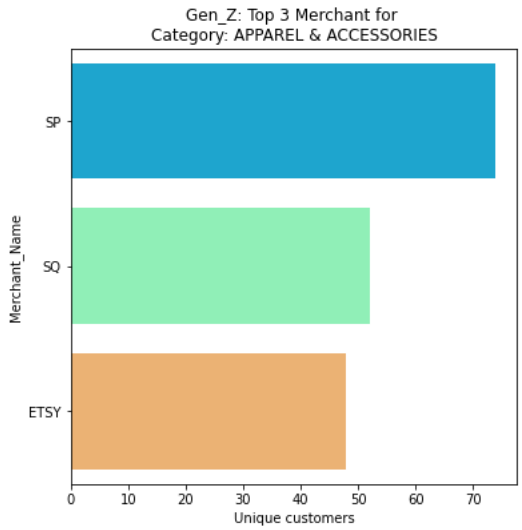

Gen_Z: Top 3 Merchant for
Category: HEALTH CARE
SQ
SANFORD
ANIMAL
Merchant_Name
0
5
10
15
20
Unique customers

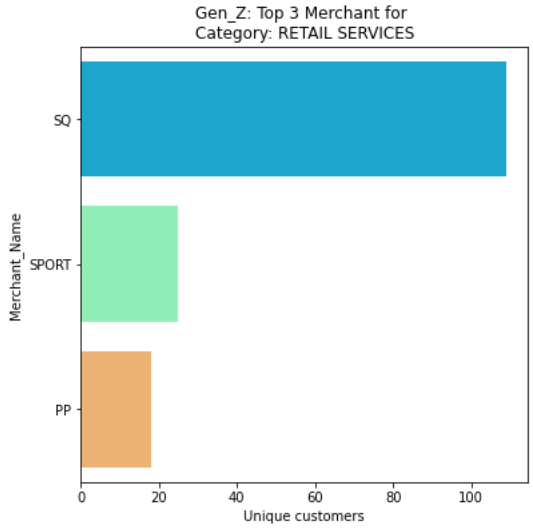

Gen_Z: Top 3 Merchant for
Category: HOME IMPROVEMENT & SUPPLY
MENARDS
LOWES
HARBOR
Merchant_Name
0
20
40
60
80
100
Unique customers

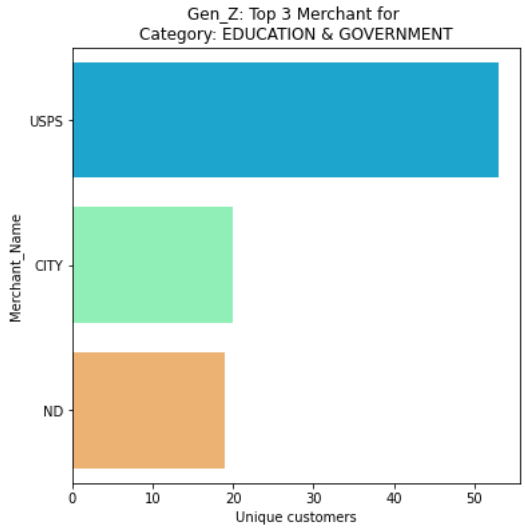

Gen_Z: Top 3 Merchant for
Category: DRUG STORES & PHARMACIES
WALGREENS
CVS/PHARMACY
RITE
Merchant_Name
0
25
50
75
100
125
150
175
200
Unique customers

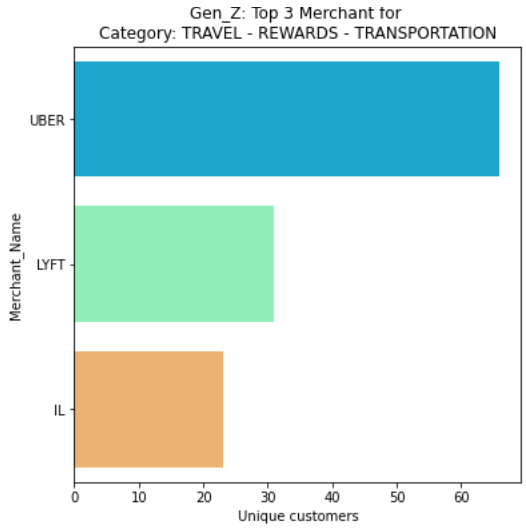

Gen_Z: Top 3 Merchant for
Category: TRAVEL - REWARDS - AIRLINES
DELTA

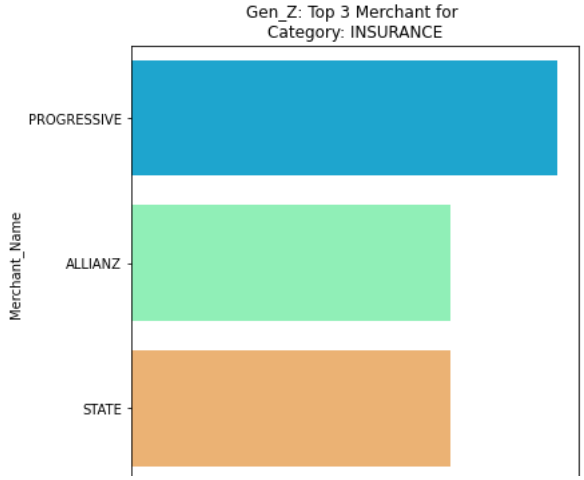

Merchant_Name
UNITED
AMERICAN
0
5
10
15
20
25
30
35
Unique customers


Gen_Z: Top 3 Merchant for
Category: APPAREL & ACCESSORIES
Merchant_Name
SP
SQ
ETSY
0
10
20
30
40
50
60
70
Unique customers


Gen_Z: Top 3 Merchant for
Category: RETAIL SERVICES
Merchant_Name
SQ
SPORT
PP
0
20
40
60
80
100
Unique customers


Gen_Z: Top 3 Merchant for
Category: EDUCATION & GOVERNMENT
Merchant_Name
USPS
CITY
ND
0
10
20
30
40
50
Unique customers



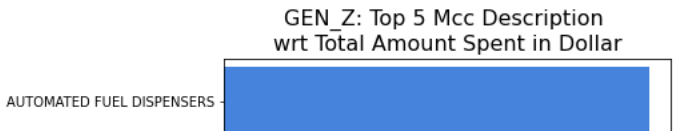


Analysis of MCC_DESRIPTION for GEN_Z

Gen_Z: Top 3 Merchant for

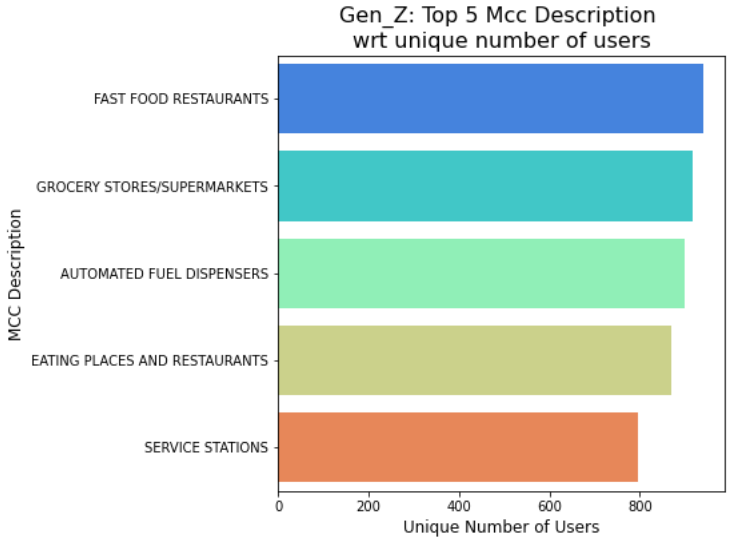
```
x=Gen_Z['MCC_DESCRIPTION'].value_counts().rename_axis('MCC_DESCRIPTION').reset_index(name='Total_Transactions')
avg_transaction = pd.DataFrame(Gen_Z.groupby('MCC_DESCRIPTION')['TRANSACTIONAMT'].mean())
total_transaction = pd.DataFrame(Gen_Z.groupby('MCC_DESCRIPTION')['TRANSACTIONAMT'].sum())
unique_customer = pd.DataFrame(Gen_Z.groupby('MCC_DESCRIPTION')['acct_id'].nunique())
merchant_avg=pd.merge(x, avg_transaction, on = 'MCC_DESCRIPTION')
merchant_all=pd.merge(merchant_avg, total_transaction, on = 'MCC_DESCRIPTION')
merchant_all=pd.merge(merchant_all, unique_customer, on = 'MCC_DESCRIPTION')
merchant_all_min_trans_30=merchant_all[merchant_all['Total_Transactions']>30]
merchant_all_min_trans_30=merchant_all_min_trans_30.rename(columns={"TRANSACTIONAMT_x": "AVG", "TRANSACTIONAMT_y": "SUM"})
Gen_Z_Top_merchant=merchant_all_min_trans_30.nlargest(n=5, columns=['SUM'])
fig=plt.figure(figsize = (6,6))
plt.title('GEN_Z: Top 5 Mcc Description \nwrt Total Amount Spent in Dollar', fontsize=16)
plt.xlabel('xlabel', fontsize=12)
plt.ylabel('ylabel', fontsize=12)
sn.barplot(x="SUM", y="MCC_DESCRIPTION", data=Gen_Z_Top_merchant,label="MCC_DESCRIPTION", palette='rainbow').set(xlabel='Amount in Dollar',yl
```

```
[Text(0, 0.5, 'MCC Description'), Text(0.5, 0, 'Amount in Dollar')]
```




```
Gen_Z_Top_merchant=merchant_all_min_trans_30.nlargest(n=5, columns=['acct_id'])
fig=plt.figure(figsize = (6,6))
fig=plt.figure(figsize = (6,6))
plt.title('Gen_Z: Top 5 Mcc Description \nwrt unique number of users', fontsize=16)
plt.xlabel('xlabel', fontsize=12)
plt.ylabel('ylabel', fontsize=12)
sn.barplot(x="acct_id", y="MCC_DESCRIPTION", data=Gen_Z_Top_merchant,label="MCC_DESCRIPTION", palette='rainbow').set(xlabel='Unique Number of
```

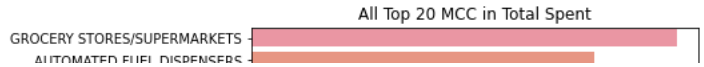
```
[Text(0, 0.5, 'MCC Description'), Text(0.5, 0, 'Unique Number of Users')]
<Figure size 432x432 with 0 Axes>
```



Double-click (or enter) to edit

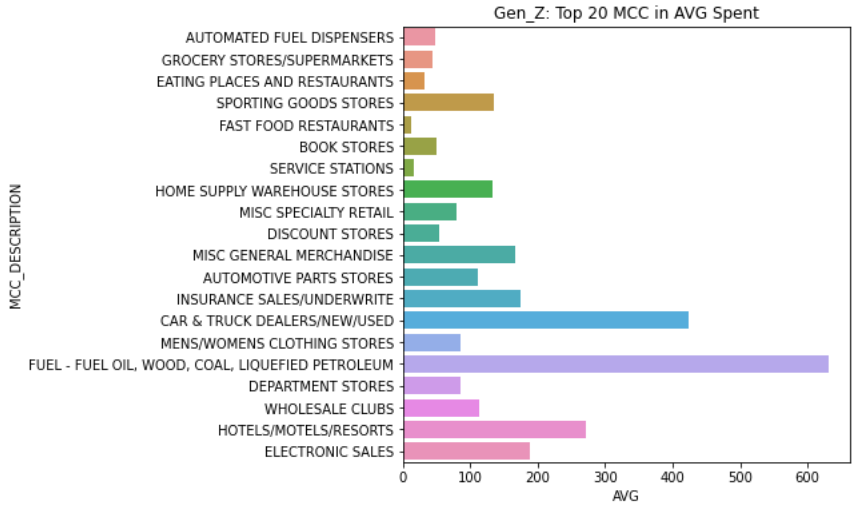
```
x=df['MCC_DESCRIPTION'].value_counts().rename_axis('MCC_DESCRIPTION').reset_index(name='Total_Transactions')
avg_transaction = pd.DataFrame(df.groupby('MCC_DESCRIPTION')['TRANSACTIONAMT'].mean())
total_transaction = pd.DataFrame(df.groupby('MCC_DESCRIPTION')['TRANSACTIONAMT'].sum())
unique_customer = pd.DataFrame(df.groupby('MCC_DESCRIPTION')['acct_id'].nunique())
merchant_avg=pd.merge(x, avg_transaction, on = 'MCC_DESCRIPTION')
merchant_all=pd.merge(merchant_avg, total_transaction, on = 'MCC_DESCRIPTION')
merchant_all=pd.merge(merchant_all, unique_customer, on = 'MCC_DESCRIPTION')
merchant_all_min_trans_30=merchant_all[merchant_all['Total_Transactions']>30]
merchant_all_min_trans_30=merchant_all_min_trans_30.rename(columns={"TRANSACTIONAMT_x": "AVG", "TRANSACTIONAMT_y": "SUM"})
Gen_Z_Top_merchant=merchant_all_min_trans_30.nlargest(n=20, columns=['SUM'])
plt.figure(figsize = (6,6))
sn.barplot(x="SUM", y="MCC_DESCRIPTION", data=Gen_Z_Top_merchant,label="MCC_DESCRIPTION").set(title='All Top 20 MCC in Total Spent')
```

```
[Text(0.5, 1.0, 'All Top 20 MCC in Total Spent')]
```



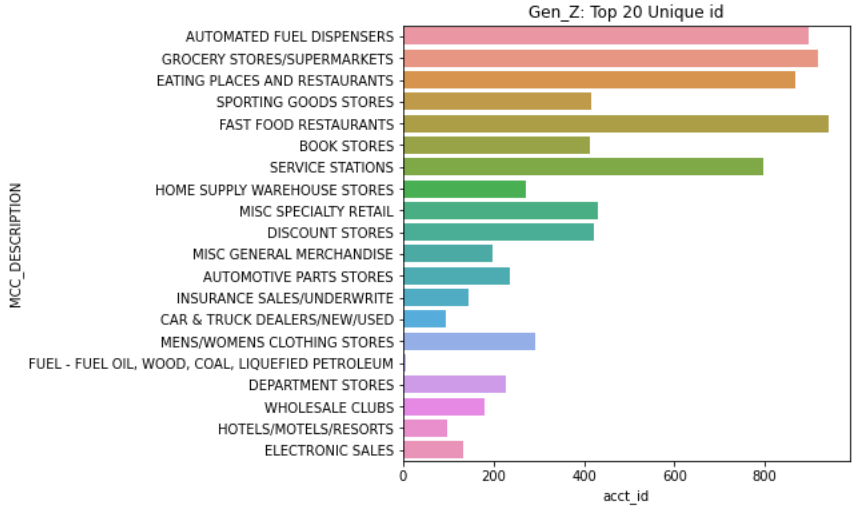

```
plt.figure(figsize = (6,6))
sn.barplot(x="AVG", y="MCC_DESCRIPTION", data=Gen_Z_Top_merchant,label="MCC_DESCRIPTION").set(title='Gen_Z: Top 20 MCC in AVG Spent')
```

```
[Text(0.5, 1.0, 'Gen_Z: Top 20 MCC in AVG Spent')]
```




```
plt.figure(figsize = (6,6))
sn.barplot(x="acct_id", y="MCC_DESCRIPTION", data=Gen_Z_Top_merchant,label="MCC_DESCRIPTION").set(title='Gen_Z: Top 20 Unique id')
```

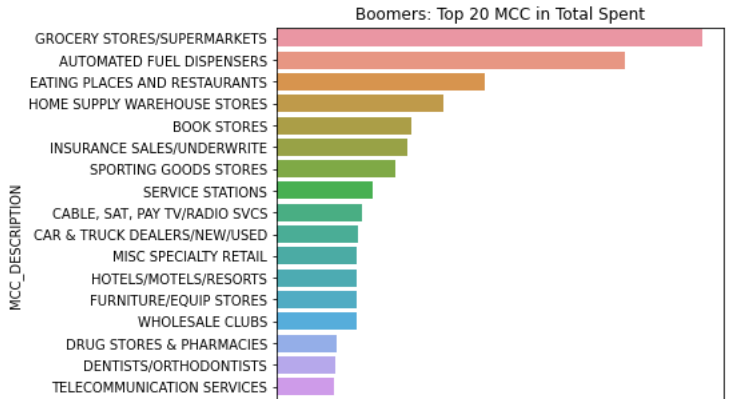
```
[Text(0.5, 1.0, 'Gen_Z: Top 20 Unique id')]
```



Analysis of MCC_DESRIPTION for Boomers

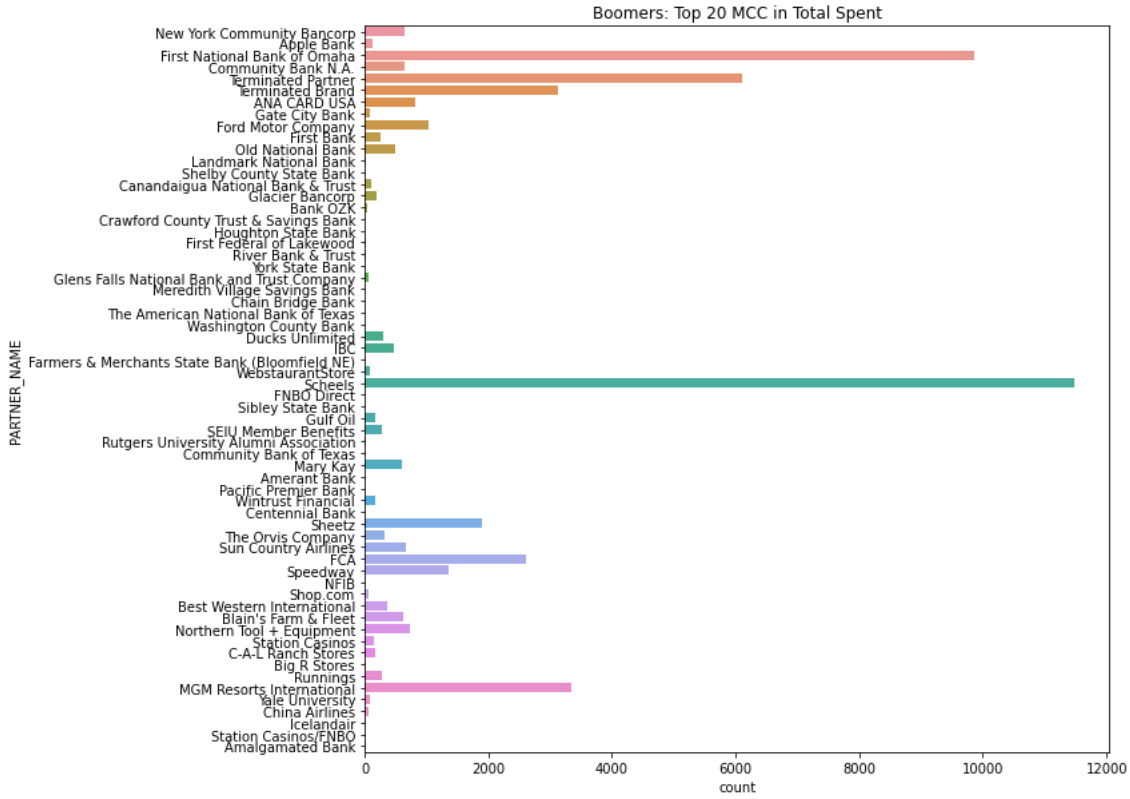
```
x=Boomers['MCC_DESCRIPTION'].value_counts().rename_axis('MCC_DESCRIPTION').reset_index(name='Total_Transactions')
avg_transaction = pd.DataFrame(Boomers.groupby('MCC_DESCRIPTION')['TRANSACTIONAMT'].mean())
total_transaction = pd.DataFrame(Boomers.groupby('MCC_DESCRIPTION')['TRANSACTIONAMT'].sum())
unique_customer = pd.DataFrame(Boomers.groupby('MCC_DESCRIPTION')['acct_id'].nunique())
merchant_avg=pd.merge(x, avg_transaction, on = 'MCC_DESCRIPTION')
merchant_all=pd.merge(merchant_avg, total_transaction, on = 'MCC_DESCRIPTION')
merchant_all=pd.merge(merchant_all, unique_customer, on = 'MCC_DESCRIPTION')
merchant_all_min_trans_30=merchant_all[merchant_all['Total_Transactions']>30]
merchant_all_min_trans_30=merchant_all_min_trans_30.rename(columns={"TRANSACTIONAMT_x": "AVG", "TRANSACTIONAMT_y": "SUM"})
Gen_Z_Top_merchant=merchant_all_min_trans_30.nlargest(n=20, columns=['SUM'])
plt.figure(figsize = (6,6))
sn.barplot(x="SUM", y="MCC_DESCRIPTION", data=Gen_Z_Top_merchant,label="MCC_DESCRIPTION").set(title='Boomers: Top 20 MCC in Total Spent')
```

```
[Text(0.5, 1.0, 'Boomers: Top 20 MCC in Total Spent')]
```



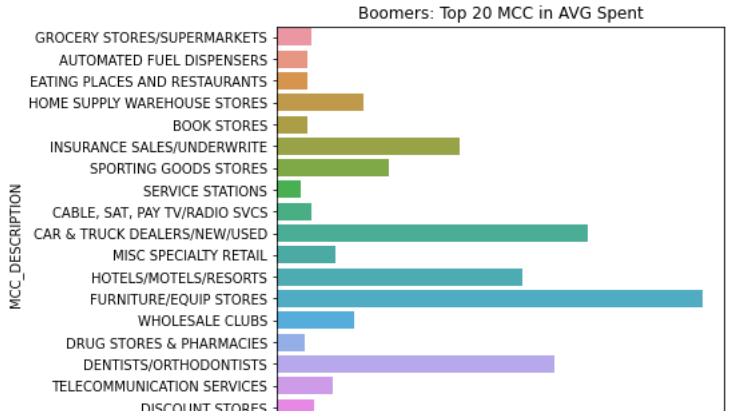

```
plt.figure(figsize = (10,10))
sn.countplot(y="PARTNER_NAME", data=cus_data,label="MCC_DESCRIPTION").set(title='Boomers: Top 20 MCC in Total Spent')
```

```
[Text(0.5, 1.0, 'Boomers: Top 20 MCC in Total Spent')]
```



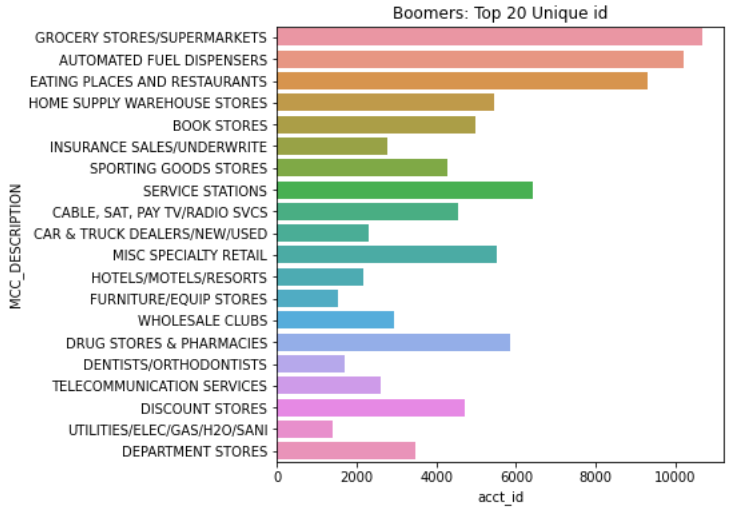

```
x=Boomers['MCC_DESCRIPTION'].value_counts().rename_axis('MCC_DESCRIPTION').reset_index(name='Total_Transactions')
avg_transaction = pd.DataFrame(Boomers.groupby('MCC_DESCRIPTION')['TRANSACTIONAMT'].mean())
total_transaction = pd.DataFrame(Boomers.groupby('MCC_DESCRIPTION')['TRANSACTIONAMT'].sum())
unique_customer = pd.DataFrame(Boomers.groupby('MCC_DESCRIPTION')['acct_id'].nunique())
merchant_avg=pd.merge(x, avg_transaction, on = 'MCC_DESCRIPTION')
merchant_all=pd.merge(merchant_avg, total_transaction, on = 'MCC_DESCRIPTION')
merchant_all=pd.merge(merchant_all, unique_customer, on = 'MCC_DESCRIPTION')
merchant_all_min_trans_30=merchant_all[merchant_all['Total_Transactions']>30]
merchant_all_min_trans_30=merchant_all_min_trans_30.rename(columns={"TRANSACTIONAMT_x": "AVG", "TRANSACTIONAMT_y": "SUM"})
Gen_Z_Top_merchant=merchant_all_min_trans_30.nlargest(n=20, columns=['SUM'])
plt.figure(figsize = (6,6))
sn.barplot(x="SUM", y="MCC_DESCRIPTION", data=Gen_Z_Top_merchant,label="MCC_DESCRIPTION").set(title='Boomers: Top 20 MCC in Total Spent')


plt.figure(figsize = (6,6))
sn.barplot(x="AVG", y="MCC_DESCRIPTION", data=Gen_Z_Top_merchant,label="MCC_DESCRIPTION").set(title='Boomers: Top 20 MCC in AVG Spent')
```

```
[Text(0.5, 1.0, 'Boomers: Top 20 MCC in AVG Spent')]
```




```
plt.figure(figsize = (6,6))
sn.barplot(x="acct_id", y="MCC_DESCRIPTION", data=Gen_Z_Top_merchant,label="MCC_DESCRIPTION").set(title='Boomers: Top 20 Unique id')
```

```
[Text(0.5, 1.0, 'Boomers: Top 20 Unique id')]
```




```
Gen_Z.columns
```

```
Index(['acct_id', 'VINTAGE_YEAR', 'STATE', 'GENERATION', 'AGE', 'FICO',
       'MONTHLY_INCOME', 'PRODUCT_SEGMENT', 'PARTNERBUSINESSKEY',
       'PARTNER_NAME', 'CREDIT_LIMIT', 'TRANSACTIONPOSTDT', 'TRANSACTIONAMT',
       'MERCHANTDESC', 'MCC', 'MCC_DESCRIPTION', 'SPENDCATEGORYNM', 'ENTRYNBR',
       'ME_BALANCE', 'date', 'buy_month', 'time_series', 'buy_weekday',
       'Merchant_Name', 'Spending_ratio'],
      dtype='object')
```

Spending Ratio

```
spending=pd.DataFrame(Gen_Z.groupby(['acct_id','buy_month'])['TRANSACTIONAMT'].sum().round(2))
spending.reset_index(inplace=True)
```

```
len(spending)
```

```
7692
```

```
Gen_Z['acct_id'].nunique()
```

```
1277
```

```
limit=pd.DataFrame(Gen_Z.groupby('acct_id')['CREDIT_LIMIT'].max())
limit.reset_index(inplace=True)
```

```
spending=spending.merge(limit, on='acct_id')
```

```
id=Gen_Z['acct_id'].unique()
for i in range(10):
  data=spending[spending['acct_id']== id[random.randint(0,1200)]]
  print(data)
```

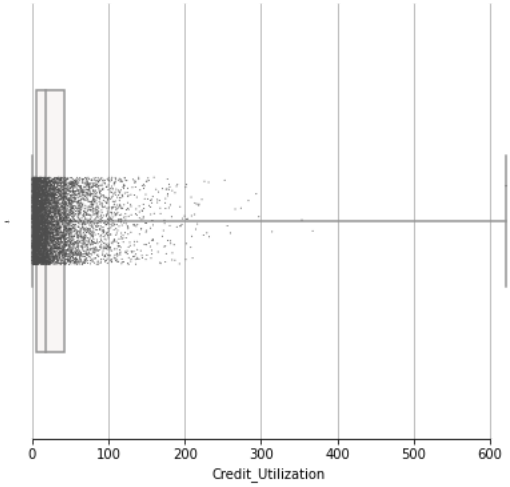
```
                acct_id  buy_month  TRANSACTIONAMT  CREDIT_LIMIT
6765  EDM0000008753475          1            33.4        1600.0
6766  EDM0000008753475          3           106.5        1600.0
                acct_id  buy_month  TRANSACTIONAMT  CREDIT_LIMIT
2459  EDM0000003536736          1          122.10         900.0
2460  EDM0000003536736          2          290.15         900.0
2461  EDM0000003536736          3          228.34         900.0
2462  EDM0000003536736          4          257.02         900.0
2463  EDM0000003536736          5          393.34         900.0
2464  EDM0000003536736          6          484.31         900.0
2465  EDM0000003536736          7          249.52         900.0
                acct_id  buy_month  TRANSACTIONAMT  CREDIT_LIMIT
3003  EDM0000003907290          1          410.50        1000.0
3004  EDM0000003907290          2          275.24        1000.0
3005  EDM0000003907290          3          487.45        1000.0
3006  EDM0000003907290          4          472.18        1000.0
3007  EDM0000003907290          5          271.40        1000.0
3008  EDM0000003907290          6          744.55        1000.0
3009  EDM0000003907290          7           13.45        1000.0
                acct_id  buy_month  TRANSACTIONAMT  CREDIT_LIMIT
6271  EDM0000008600049          1           46.70         500.0
6272  EDM0000008600049          5          629.70         500.0
6273  EDM0000008600049          6          256.74         500.0
6274  EDM0000008600049          7            2.16         500.0
                acct_id  buy_month  TRANSACTIONAMT  CREDIT_LIMIT
6712  EDM0000008732375          1          152.03        9400.0
6713  EDM0000008732375          2         1371.79        9400.0
6714  EDM0000008732375          3          331.52        9400.0
6715  EDM0000008732375          4            8.82        9400.0
                acct_id  buy_month  TRANSACTIONAMT  CREDIT_LIMIT
6250  EDM0000008595453          1          433.25        1000.0
6251  EDM0000008595453          2          309.94        1000.0
6252  EDM0000008595453          3          425.30        1000.0
6253  EDM0000008595453          4          336.31        1000.0
6254  EDM0000008595453          5          101.39        1000.0
6255  EDM0000008595453          6          521.73        1000.0
6256  EDM0000008595453          7          553.94        1000.0
                acct_id  buy_month  TRANSACTIONAMT  CREDIT_LIMIT
6257  EDM0000008596634          1          144.90        1000.0
6258  EDM0000008596634          2          352.54        1000.0
6259  EDM0000008596634          3          277.68        1000.0
6260  EDM0000008596634          4           66.84        1000.0
6261  EDM0000008596634          5          387.01        1000.0
6262  EDM0000008596634          6          861.20        1000.0
6263  EDM0000008596634          7          301.95        1000.0
                acct_id  buy_month  TRANSACTIONAMT  CREDIT_LIMIT
6297  EDM0000008602752          1          215.86        1500.0
6298  EDM0000008602752          2           61.00        1500.0
6299  EDM0000008602752          3          362.36        1500.0
6300  EDM0000008602752          4          292.76        1500.0
6301  EDM0000008602752          5           80.09        1500.0
6302  EDM0000008602752          6         1352.30        1500.0
6303  EDM0000008602752          7          323.57        1500.0
                acct_id  buy_month  TRANSACTIONAMT  CREDIT_LIMIT
5555  EDM0000006696916          1           54.00        1400.0
5556  EDM0000006696916          2           15.12        1400.0
5557  EDM0000006696916          7          123.29        1400.0
                acct_id  buy_month  TRANSACTIONAMT  CREDIT_LIMIT
```

```
spending['Credit_Utilization']=((spending['TRANSACTIONAMT']/spending['CREDIT_LIMIT'])*100).round(2)
```

```
spending.take([103,104,105,106,107,108])
```

	acct_id	buy_month	TRANSACTIONAMT	CREDIT_LIMIT	Credit_Utilization
103	EDM0000000376174	1	726.70	5000.0	14.53
104	EDM0000000376174	2	1035.33	5000.0	20.71
105	EDM0000000376174	3	1610.14	5000.0	32.20
106	EDM0000000376174	4	1475.95	5000.0	29.52
107	EDM0000000376174	5	461.39	5000.0	9.23
108	EDM0000000376174	6	858.26	5000.0	17.17

```
#sns.set_theme(style="ticks")

# Initialize the figure with a logarithmic x axis
f, ax = plt.subplots(figsize=(7, 6))
#ax.set_xscale("log")

# Load the example planets dataset
#planets = sns.load_dataset("planets")

# Plot the orbital period with horizontal boxes
sn.boxplot(x="Credit_Utilization", data=spending, whis=[0, 100], width=.6, palette="vlag")

# Add in points to show each observation
sn.stripplot(x="Credit_Utilization", data=spending, size=1, color=".3", linewidth=0)

# Tweak the visual presentation
ax.xaxis.grid(True)
ax.set(ylabel="")
sn.despine(trim=True, left=True)
```

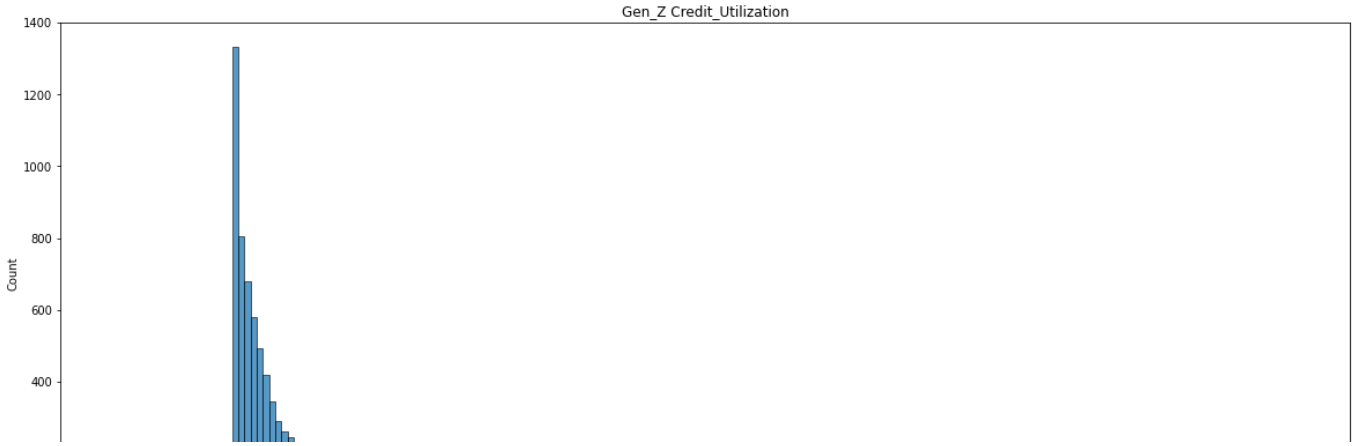



```
f, ax = plt.subplots(figsize=(20, 8))
sn.histplot(x='Credit_Utilization', data = spending).set(title='Gen_Z Credit_Utilization')
ax.set_xticks([i for i in range(-100,650,10)])
```

```
[<matplotlib.axis.XTick at 0x7fda389342d0>,
 <matplotlib.axis.XTick at 0x7fda38934350>,
 <matplotlib.axis.XTick at 0x7fda38991490>,
 <matplotlib.axis.XTick at 0x7fda38b30110>,
 <matplotlib.axis.XTick at 0x7fda38b86490>,
 <matplotlib.axis.XTick at 0x7fda38b7b710>,
 <matplotlib.axis.XTick at 0x7fda38b76c50>,
 <matplotlib.axis.XTick at 0x7fda38b71b50>,
 <matplotlib.axis.XTick at 0x7fda38b221d0>,
 <matplotlib.axis.XTick at 0x7fda38936f50>,
 <matplotlib.axis.XTick at 0x7fda3898ebd0>,
 <matplotlib.axis.XTick at 0x7fda3898f510>,
 <matplotlib.axis.XTick at 0x7fda38b5ea10>,
 <matplotlib.axis.XTick at 0x7fda38b6ed50>,
 <matplotlib.axis.XTick at 0x7fda38b6d990>,
 <matplotlib.axis.XTick at 0x7fda38bc12d0>,
 <matplotlib.axis.XTick at 0x7fda38bbc8d0>,
 <matplotlib.axis.XTick at 0x7fda38bae5d0>,
 <matplotlib.axis.XTick at 0x7fda38ba0210>,
 <matplotlib.axis.XTick at 0x7fda38b9ae50>,
 <matplotlib.axis.XTick at 0x7fda38baea50>,
 <matplotlib.axis.XTick at 0x7fda38b9b490>,
 <matplotlib.axis.XTick at 0x7fda38bc8c50>,
 <matplotlib.axis.XTick at 0x7fda38b96bd0>,
 <matplotlib.axis.XTick at 0x7fda38b95310>,
 <matplotlib.axis.XTick at 0x7fda38c10310>,
 <matplotlib.axis.XTick at 0x7fda38b956d0>,
 <matplotlib.axis.XTick at 0x7fda38c017d0>,
 <matplotlib.axis.XTick at 0x7fda38c001d0>,
 <matplotlib.axis.XTick at 0x7fda38bff550>,
 <matplotlib.axis.XTick at 0x7fda38c09690>,
 <matplotlib.axis.XTick at 0x7fda38bc1150>,
 <matplotlib.axis.XTick at 0x7fda3898f4d0>,
 <matplotlib.axis.XTick at 0x7fda38bfe3d0>,
 <matplotlib.axis.XTick at 0x7fda38bfee10>,
 <matplotlib.axis.XTick at 0x7fda38be37d0>,
 <matplotlib.axis.XTick at 0x7fda38bd5550>,
 <matplotlib.axis.XTick at 0x7fda38bd3090>,
 <matplotlib.axis.XTick at 0x7fda38c3ef10>,
 <matplotlib.axis.XTick at 0x7fda38c36b90>,
 <matplotlib.axis.XTick at 0x7fda38bd5950>,
 <matplotlib.axis.XTick at 0x7fda38bfe550>,
 <matplotlib.axis.XTick at 0x7fda38c35f90>,
 <matplotlib.axis.XTick at 0x7fda38c346d0>,
 <matplotlib.axis.XTick at 0x7fda38c2e390>,
 <matplotlib.axis.XTick at 0x7fda38c2ce10>,
 <matplotlib.axis.XTick at 0x7fda38c21650>,
 <matplotlib.axis.XTick at 0x7fda38c138d0>,
 <matplotlib.axis.XTick at 0x7fda38c13190>,
 <matplotlib.axis.XTick at 0x7fda38c13890>,
 <matplotlib.axis.XTick at 0x7fda38c34b50>,
 <matplotlib.axis.XTick at 0x7fda38c3df50>,
 <matplotlib.axis.XTick at 0x7fda38c12f10>,
 <matplotlib.axis.XTick at 0x7fda38c91210>,
 <matplotlib.axis.XTick at 0x7fda38c90350>,
 <matplotlib.axis.XTick at 0x7fda38c8f450>,
 <matplotlib.axis.XTick at 0x7fda38c8e610>,
 <matplotlib.axis.XTick at 0x7fda38c8ddd0>,
 <matplotlib.axis.XTick at 0x7fda38c8b690>,
 <matplotlib.axis.XTick at 0x7fda38c8cb10>,
 <matplotlib.axis.XTick at 0x7fda38c90d90>,
 <matplotlib.axis.XTick at 0x7fda38c2ce90>,
 <matplotlib.axis.XTick at 0x7fda38c8a350>,
 <matplotlib.axis.XTick at 0x7fda38c88c90>,
 <matplotlib.axis.XTick at 0x7fda38c727d0>,
 <matplotlib.axis.XTick at 0x7fda38c6eb50>,
 <matplotlib.axis.XTick at 0x7fda38c66a90>,
 <matplotlib.axis.XTick at 0x7fda38c65150>,
```

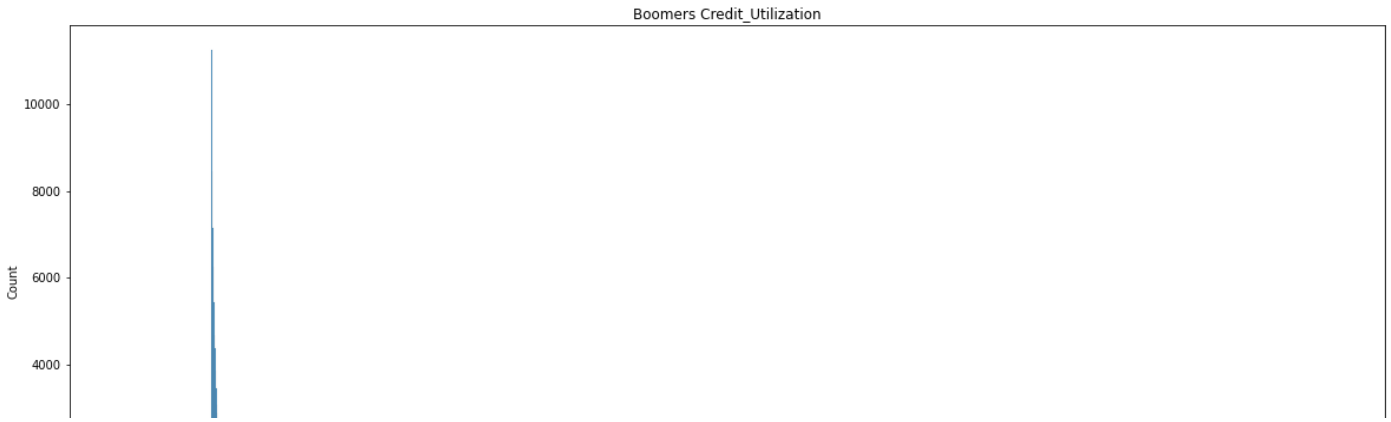
```
spending=pd.DataFrame(Gen_Z.groupby(['acct_id','buy_month'])['TRANSACTIONAMT'].sum().round(2))
spending.reset_index(inplace=True)
limit=pd.DataFrame(Gen_Z.groupby('acct_id')['CREDIT_LIMIT'].max())
limit.reset_index(inplace=True)
spending=spending.merge(limit, on='acct_id')
spending['Credit_Utilization']=((spending['TRANSACTIONAMT']/spending['CREDIT_LIMIT'])*100).round(2)
f, ax = plt.subplots(figsize=(20, 8))
sn.histplot(x='Credit_Utilization', data = spending).set(title='Gen_Z Credit_Utilization')
ax.set_xticks([i for i in range(-100,650,20)])
```

```
[<matplotlib.axis.XTick at 0x7fda38b07bd0>,
 <matplotlib.axis.XTick at 0x7fda38ce0bd0>,
 <matplotlib.axis.XTick at 0x7fda38d18290>,
 <matplotlib.axis.XTick at 0x7fda38f71510>,
 <matplotlib.axis.XTick at 0x7fda38f55150>,
 <matplotlib.axis.XTick at 0x7fda38f531d0>,
 <matplotlib.axis.XTick at 0x7fda38fd1a50>,
 <matplotlib.axis.XTick at 0x7fda38fc0950>,
 <matplotlib.axis.XTick at 0x7fda38fc0990>,
 <matplotlib.axis.XTick at 0x7fda38511350>,
 <matplotlib.axis.XTick at 0x7fda38d89690>,
 <matplotlib.axis.XTick at 0x7fda38d8ab50>,
 <matplotlib.axis.XTick at 0x7fda38d8bc50>,
 <matplotlib.axis.XTick at 0x7fda38fb4a10>,
 <matplotlib.axis.XTick at 0x7fda38fb1bd0>,
 <matplotlib.axis.XTick at 0x7fda38d91a10>,
 <matplotlib.axis.XTick at 0x7fda38fb6f10>,
 <matplotlib.axis.XTick at 0x7fda38d8abd0>,
 <matplotlib.axis.XTick at 0x7fda38f53e90>,
 <matplotlib.axis.XTick at 0x7fda38f9e910>,
 <matplotlib.axis.XTick at 0x7fda38f9ea90>,
 <matplotlib.axis.XTick at 0x7fda38f99c10>,
 <matplotlib.axis.XTick at 0x7fda38f95850>,
 <matplotlib.axis.XTick at 0x7fda38f94d50>,
 <matplotlib.axis.XTick at 0x7fda390112d0>,
 <matplotlib.axis.XTick at 0x7fda38f94c10>,
 <matplotlib.axis.XTick at 0x7fda38f9ecd0>,
 <matplotlib.axis.XTick at 0x7fda38d8acd0>,
 <matplotlib.axis.XTick at 0x7fda39007150>,
 <matplotlib.axis.XTick at 0x7fda39006ed0>,
 <matplotlib.axis.XTick at 0x7fda39005410>,
 <matplotlib.axis.XTick at 0x7fda39001b50>,
 <matplotlib.axis.XTick at 0x7fda38fffa50>,
 <matplotlib.axis.XTick at 0x7fda39000d90>,
 <matplotlib.axis.XTick at 0x7fda38ffb810>,
 <matplotlib.axis.XTick at 0x7fda38fff8d0>,
 <matplotlib.axis.XTick at 0x7fda38ffbd90>,
 <matplotlib.axis.XTick at 0x7fda38fb5610>]
```




```
spending=pd.DataFrame(Boomers.groupby(['acct_id','buy_month'])['TRANSACTIONAMT'].sum().round(2))
spending.reset_index(inplace=True)
limit=pd.DataFrame(Boomers.groupby('acct_id')['CREDIT_LIMIT'].mean())
limit.reset_index(inplace=True)
spending=spending.merge(limit, on='acct_id')
spending['Credit_Utilization']=((spending['TRANSACTIONAMT']/spending['CREDIT_LIMIT'])*100).round(2)
f, ax = plt.subplots(figsize=(20, 8))
sn.histplot(x='Credit_Utilization', data = spending).set(title='Boomers Credit_Utilization')
ax.set_xticks([i for i in range(-100,200,20)])
```

```
[<matplotlib.axis.XTick at 0x7fda38a519d0>,
 <matplotlib.axis.XTick at 0x7fda389e0910>,
 <matplotlib.axis.XTick at 0x7fda3914dd90>,
 <matplotlib.axis.XTick at 0x7fda6f174410>,
 <matplotlib.axis.XTick at 0x7fda37986ad0>,
 <matplotlib.axis.XTick at 0x7fda379c4590>,
 <matplotlib.axis.XTick at 0x7fda379c6610>,
 <matplotlib.axis.XTick at 0x7fda379c6290>,
 <matplotlib.axis.XTick at 0x7fda37986890>,
 <matplotlib.axis.XTick at 0x7fda37918c90>,
 <matplotlib.axis.XTick at 0x7fda379c82d0>,
 <matplotlib.axis.XTick at 0x7fda38c63f50>,
 <matplotlib.axis.XTick at 0x7fda38883a10>,
 <matplotlib.axis.XTick at 0x7fda3883b610>,
 <matplotlib.axis.XTick at 0x7fda38b9b050>]
```



Credit Utilization based on age

```
data=df[((df['AGE']<31 ) | (df['GENERATION']=='2. Boomers')) & (df['TRANSACTIONAMT']>0 )]
#data=df[(df['AGE']<31 ) ]
```

```
spending=pd.DataFrame(data.groupby(['acct_id','buy_month'])['TRANSACTIONAMT'].sum().round(2))
spending.reset_index(inplace=True)
```

```
avg_spending=pd.DataFrame(data.groupby('acct_id')['TRANSACTIONAMT'].mean().round(2))
avg_spending.reset_index(inplace=True)
```

```
avg_spending.head()
```

	acct_id	TRANSACTIONAMT
0	EDM0000000002119	98.12
1	EDM0000000007360	27.90
2	EDM0000000008180	52.57
3	EDM0000000018628	15.21
4	EDM0000000022348	108.10

```
cus_data=data[['acct_id','AGE', 'CREDIT_LIMIT','FICO','MONTHLY_INCOME']]
cus_data.drop_duplicates( "acct_id" , keep='first',inplace=True)
avg_spending=avg_spending.merge(cus_data, on='acct_id')
```

```
/usr/local/lib/python3.7/dist-packages/pandas/util/_decorators.py:311: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
  return func(*args, **kwargs)
```

```
df.columns
```

```
Index(['acct_id', 'VINTAGE_YEAR', 'STATE', 'GENERATION', 'AGE', 'FICO',
       'MONTHLY_INCOME', 'PRODUCT_SEGMENT', 'PARTNERBUSINESSKEY',
       'PARTNER_NAME', 'CREDIT_LIMIT', 'TRANSACTIONPOSTDT', 'TRANSACTIONAMT',
       'MERCHANTDESC', 'MCC', 'MCC_DESCRIPTION', 'SPENDCATEGORYNM', 'ENTRYNBR',
```

```
       'ME_BALANCE', 'date', 'buy_month', 'time_series', 'buy_weekday'],
      dtype='object')
```

```
cus_data=df[['acct_id', 'PARTNER_NAME']]
cus_data.drop_duplicates( "acct_id" , keep='first',inplace=True)
```

```
/usr/local/lib/python3.7/dist-packages/pandas/util/_decorators.py:311: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
  return func(*args, **kwargs)
```

```
/usr/local/lib/python3.7/dist-packages/pandas/util/_decorators.py:311: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
  return func(*args, **kwargs)
```

```
avg_spending[avg_spending['acct_id']=='EDM0000000328432']
```

	acct_id	TRANSACTIONAMT	AGE	CREDIT_LIMIT	FICO	MONTHLY_INCOME
84	EDM0000000328432	18.91	26	400.0	785	4166

```
avg_spending['Credit_Utilization']=((avg_spending['TRANSACTIONAMT']/avg_spending['CREDIT_LIMIT'])*100).round(2)
```

```
avg_spending.head()
```

	acct_id	TRANSACTIONAMT	AGE	CREDIT_LIMIT	FICO	MONTHLY_INCOME	Credit_Utilization
0	EDM0000000002119	98.12	66	19250.0	799	2000	0.51
1	EDM0000000007360	27.90	68	9000.0	741	4666	0.31
2	EDM0000000008180	52.57	73	2500.0	726	3333	2.10
3	EDM0000000018628	15.21	59	2800.0	642	3750	0.54
4	EDM0000000022348	108.10	76	24650.0	734	5250	0.44

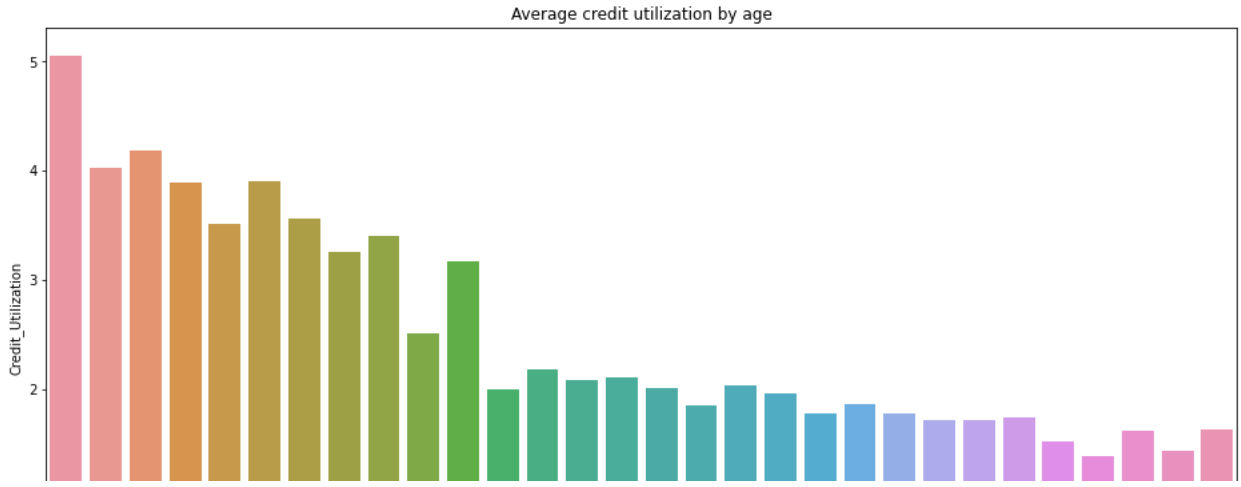
```
age_avg_utilization = pd.DataFrame(avg_spending.groupby('AGE')['Credit_Utilization'].mean())
age_avg_utilization.reset_index(inplace=True)
```

```
age_avg_utilization.head()
```

	AGE	Credit_Utilization
0	20	5.057222
1	21	4.031228
2	22	4.191824
3	23	3.895426
4	24	3.516524

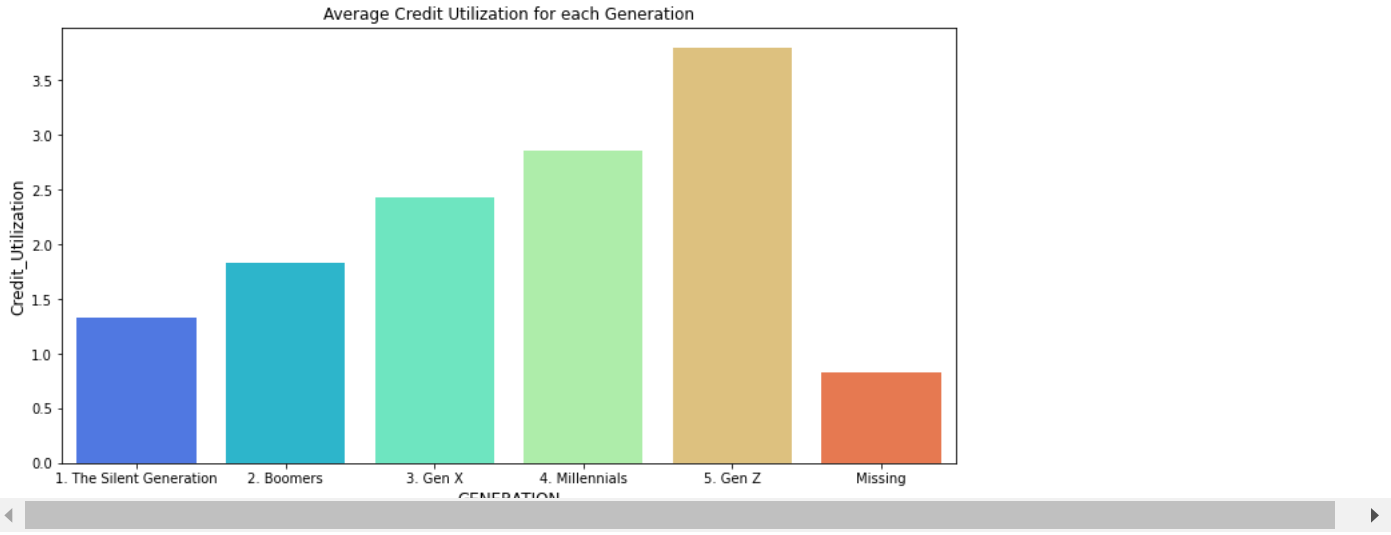
```
plt.figure(figsize = (16,8))
sn.barplot(x="AGE", y="Credit_Utilization", data=age_avg_utilization).set(title='Average credit utilization by age')
```

```
[Text(0.5, 1.0, 'Average credit utilization by age')]
```




```
data=df[(df['TRANSACTIONAMT']>0 )]
spending=pd.DataFrame(data.groupby(['acct_id','buy_month'])['TRANSACTIONAMT'].sum().round(2))
spending.reset_index(inplace=True)
avg_spending=pd.DataFrame(data.groupby('acct_id')['TRANSACTIONAMT'].mean().round(2))
avg_spending.reset_index(inplace=True)
cus_data=data[['acct_id','AGE', 'CREDIT_LIMIT','FICO','MONTHLY_INCOME','GENERATION']]
cus_data.drop_duplicates( "acct_id" , keep='first',inplace=True)
avg_spending=avg_spending.merge(cus_data, on='acct_id')
avg_spending['Credit_Utilization']=((avg_spending['TRANSACTIONAMT']/avg_spending['CREDIT_LIMIT'])*100).round(2)
age_avg_utilization = pd.DataFrame(avg_spending.groupby('GENERATION')['Credit_Utilization'].mean())
age_avg_utilization.reset_index(inplace=True)
fig=plt.figure(figsize = (12,6))
plt.title('Gen_Z: Top 3 Sporting Goods\n wrt unique number of users', fontsize=16)
plt.xlabel('xlabel', fontsize=12)
plt.ylabel('ylabel', fontsize=12)
sn.barplot(x="GENERATION", y="Credit_Utilization", data=age_avg_utilization, palette='rainbow').set(title='Average Credit Utilization for eacl
```

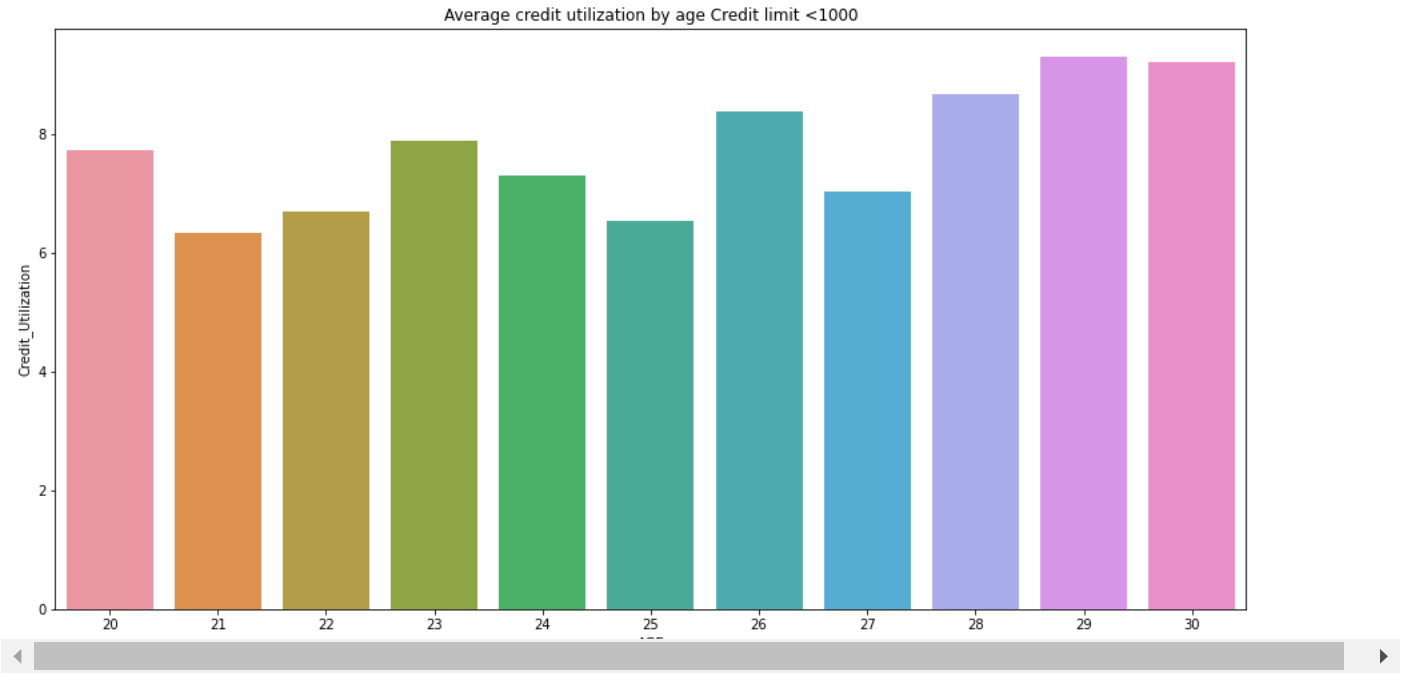
```
/usr/local/lib/python3.7/dist-packages/pandas/util/_decorators.py:311: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
  return func(*args, **kwargs)
[Text(0.5, 1.0, 'Average Credit Utilization for each Generation')]
```




```
data=df[((df['AGE']<31 )  & (df['CREDIT_LIMIT']<1000) )& (df['TRANSACTIONAMT']>0 ) ]
spending=pd.DataFrame(data.groupby(['acct_id','buy_month'])['TRANSACTIONAMT'].sum().round(2))
spending.reset_index(inplace=True)
avg_spending=pd.DataFrame(data.groupby('acct_id')['TRANSACTIONAMT'].mean().round(2))
avg_spending.reset_index(inplace=True)
cus_data=data[['acct_id','AGE', 'CREDIT_LIMIT','FICO','MONTHLY_INCOME']]
cus_data.drop_duplicates( "acct_id" , keep='first',inplace=True)
avg_spending=avg_spending.merge(cus_data, on='acct_id')
avg_spending['Credit_Utilization']=((avg_spending['TRANSACTIONAMT']/avg_spending['CREDIT_LIMIT'])*100).round(2)
age_avg_utilization = pd.DataFrame(avg_spending.groupby('AGE')['Credit_Utilization'].mean())
age_avg_utilization.reset_index(inplace=True)
```

```
plt.figure(figsize = (16,8))
sn.barplot(x="AGE", y="Credit_Utilization", data=age_avg_utilization).set(title='Average credit utilization by age Credit limit <1000')
```

```
/usr/local/lib/python3.7/dist-packages/pandas/util/_decorators.py:311: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
  return func(*args, **kwargs)
[Text(0.5, 1.0, 'Average credit utilization by age Credit limit <1000')]
```




```
data=df[((df['AGE']<31 )  & (df['FICO']<650))& (df['TRANSACTIONAMT']>0 )  ]
spending=pd.DataFrame(data.groupby(['acct_id','buy_month'])['TRANSACTIONAMT'].sum().round(2))
spending.reset_index(inplace=True)
avg_spending=pd.DataFrame(data.groupby('acct_id')['TRANSACTIONAMT'].mean().round(2))
avg_spending.reset_index(inplace=True)
cus_data=data[['acct_id','AGE', 'CREDIT_LIMIT','FICO','MONTHLY_INCOME']]
cus_data.drop_duplicates( "acct_id" , keep='first',inplace=True)
avg_spending=avg_spending.merge(cus_data, on='acct_id')
avg_spending['Credit_Utilization']=((avg_spending['TRANSACTIONAMT']/avg_spending['CREDIT_LIMIT'])*100).round(2)
age_avg_utilization = pd.DataFrame(avg_spending.groupby('AGE')['Credit_Utilization'].mean())
age_avg_utilization.reset_index(inplace=True)
plt.figure(figsize = (16,8))
sn.barplot(x="AGE", y="Credit_Utilization", data=age_avg_utilization).set(title='Average credit utilization by age FICO <650')
```