```
import pandas as pd

path1='/content/USER_TAKEHOME.csv'
users_df=pd.read_csv(path1)
users_df.head(5)
```

	ID	CREATED_DATE	BIRTH_DATE	STATE	LANGUAGE	GENDER
0	5ef3b4f17053ab141787697d	2020-06-24 20:17:54.000 Z	2000-08-11 00:00:00.000 Z	CA	es-419	female
1	5ff220d383fcfc12622b96bc	2021-01-03 19:53:55.000 Z	2001-09-24 04:00:00.000 Z	PA	en	female
2	6477950aa55bb77a0e27ee10	2023-05-31 18:42:18.000 Z	1994-10-28 00:00:00.000 Z	FL	es-419	female
3	658a306e99b40f103b63ccf8	2023-12-26 01:46:22.000 Z	NaN	NC	en	NaN
4	653cf5d6a225ea102b7ecdc2	2023-10-28 11:51:50.000 Z	1972-03-19 00:00:00.000 Z	PA	en	female

Next steps: Generate code with users_df View recommended plots New interactive sheet

```
import pandas as pd

path2='/content/TRANSACTION_TAKEHOME.csv'
transactions_df=pd.read_csv(path2)
transactions_df.head(5)
```

	RECEIPT_ID	PURCHASE_DATE	SCAN_DATE	STORE_NAME	USER_ID	BARCODE	FINAL_QUANTITY	FINAL_SALE
0	0000d256-4041-4a3e-adc4-5623fb6e0c99	2024-08-21	2024-08-21 14:19:06.539 Z	WALMART	63b73a7f3d310dceeabd4758	1.530001e+10	1.00	
1	0001455d-7a92-4a7b-a1d2-c747af1c8fd3	2024-07-20	2024-07-20 09:50:24.206 Z	ALDI	62c08877baa38d1a1f6c211a	NaN	zero	1.49
2	00017e0a-7851-42fb-bfab-	2024-08-18	2024-08-19	WALMART	60842f207ac8b7729e472020	7.874223e+10	1.00	

Next steps: Generate code with transactions_df View recommended plots New interactive sheet

```
import pandas as pd

path3='/content/PRODUCTS_TAKEHOME.csv'
products_df=pd.read_csv(path3)
products_df.head(5)
```

	CATEGORY_1	CATEGORY_2	CATEGORY_3	CATEGORY_4	MANUFACTURER	BRAND	BARCODE
0	Health & Wellness	Sexual Health	Conductivity Gels & Lotions	NaN	NaN	NaN	7.964944e+11
1	Snacks	Puffed Snacks	Cheese Curls & Puffs	NaN	NaN	NaN	2.327801e+10
2	Health & Wellness	Hair Care	Hair Care Accessories	NaN	PLACEHOLDER MANUFACTURER	ELECSOP	4.618178e+11

```
user_transactions = pd.merge(transactions_df, users_df, left_on='USER_ID', right_on='ID', how='inner')
full_data = pd.merge(user_transactions, products_df, on='BARCODE', how='inner')
```

```
full_data['FINAL_SALE'] = pd.to_numeric(full_data['FINAL_SALE'], errors='coerce')

top_categories = (
    full_data.groupby('CATEGORY_1')['FINAL_SALE']
    .sum()
    .sort_values(ascending=False)
    .head(10)
)
```

```
print(top_categories)
```

```
CATEGORY_1
Health & Wellness        229244.64
Snacks                   161387.89
Restaurant                 3377.22
Alcohol                    2816.20
Beverages                  1206.66
Dairy                      1001.24
Apparel & Accessories       595.98
Pantry                      403.31
Deli & Bakery               297.99
Name: FINAL_SALE, dtype: float64
```

Start coding or generate with AI.

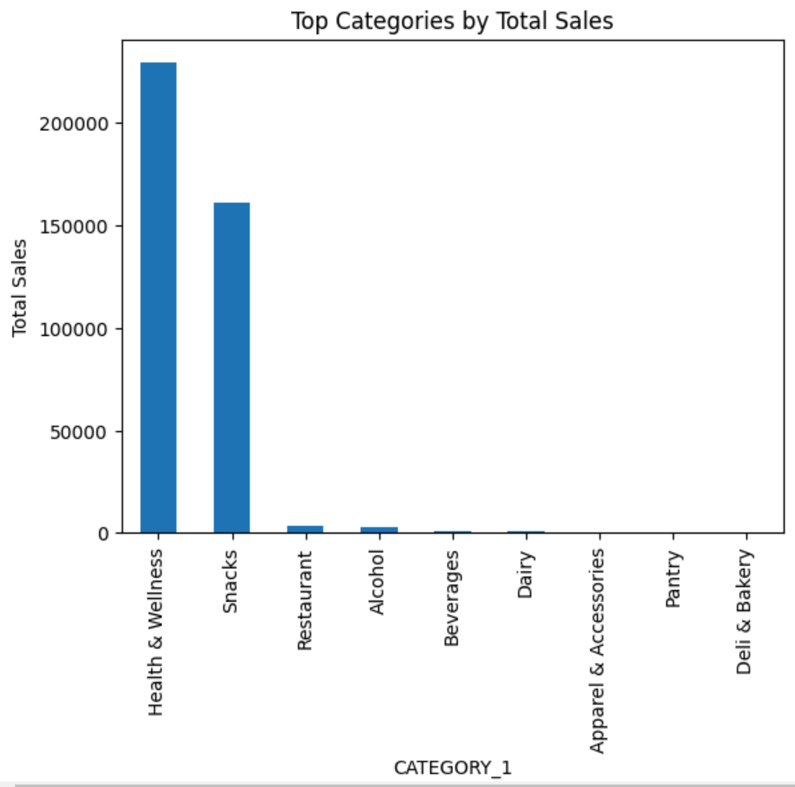
```
top_brands = (
    full_data['BRAND']
    .value_counts()
    .head(10)
)
print(top_brands)
```

```
BRAND
COCA-COLA                   628
ANNIE'S HOMEGROWN GROCERY   576
DOVE                        558
BAREFOOT                    552
ORIBE                       504
AVEENO                      480
SHEA MOISTURE               480
REESE'S                     458
NEUTROGENA                  456
FIRST AID BEAUTY            456
Name: count, dtype: int64
```

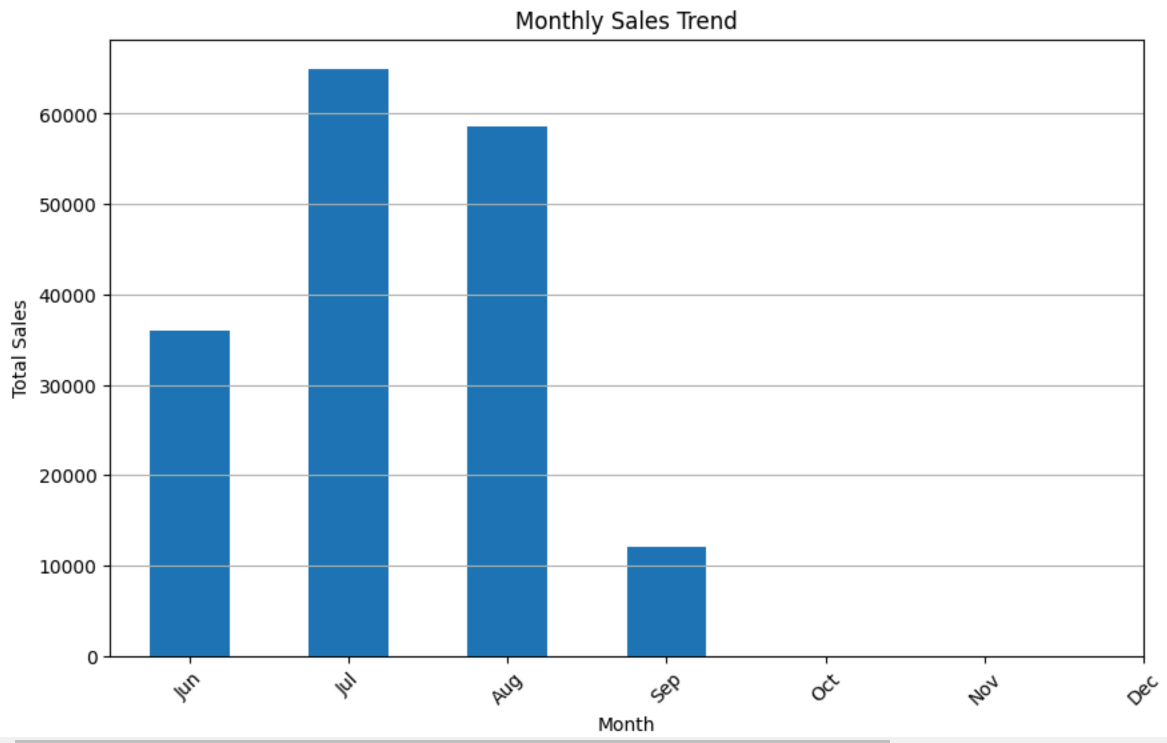
```
gender_preferences = (
    full_data.groupby(['GENDER', 'CATEGORY_1'])['FINAL_SALE']
    .sum()
    .unstack()
    .fillna(0)
)
print(gender_preferences)
```

```
CATEGORY_1  Alcohol  Apparel & Accessories  Beverages   Dairy  Deli & Bakery  \
GENDER
female      2575.96                 544.50    1103.70  915.44         272.25
male         240.24                  51.48     102.96   85.80          25.74

CATEGORY_1  Health & Wellness  Pantry  Restaurant     Snacks
GENDER
female              209361.98  368.99     3085.50  147428.74
male                 19882.66   34.32      291.72   13959.15
```

```
import matplotlib.pyplot as plt
import seaborn as sns

top_categories.plot(kind='bar', title='Top Categories by Total Sales')
plt.ylabel('Total Sales')
plt.show()
```



```
import matplotlib.pyplot as plt
import pandas as pd

transactions_df['PURCHASE_DATE'] = pd.to_datetime(transactions_df['PURCHASE_DATE'], errors='coerce')

transactions_df['FINAL_SALE'] = pd.to_numeric(transactions_df['FINAL_SALE'], errors='coerce')

transactions_df = transactions_df.dropna(subset=['FINAL_SALE'])

transactions_df['MONTH'] = transactions_df['PURCHASE_DATE'].dt.month
monthly_sales = transactions_df.groupby('MONTH')['FINAL_SALE'].sum()

monthly_sales.plot(kind='bar', title='Monthly Sales Trend', figsize=(10, 6))
plt.xlabel('Month')
plt.ylabel('Total Sales')
plt.xticks(range(7), ['Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'], rotation=45)
plt.grid(axis='y')
plt.show()
```



```
transactions_df['PURCHASE_DATE'] = pd.to_datetime(transactions_df['PURCHASE_DATE'], errors='coerce')

max_transaction_date = transactions_df['PURCHASE_DATE'].max()
min_transaction_date = transactions_df['PURCHASE_DATE'].min()


print("Maximum Transaction Date:", max_transaction_date)
print("Minimum Transaction Date:", min_transaction_date)
print(monthly_sales)
```

```
Maximum Transaction Date: 2024-09-08 00:00:00
Minimum Transaction Date: 2024-06-12 00:00:00
MONTH
6    36024.17
7    64987.69
8    58554.53
9    12048.01
Name: FINAL_SALE, dtype: float64
```

```
import pandas as pd

# Convert BIRTH_DATE and PURCHASE_DATE columns to datetime format
users_df['BIRTH_DATE'] = pd.to_datetime(users_df['BIRTH_DATE'], errors='coerce')
transactions_df['PURCHASE_DATE'] = pd.to_datetime(transactions_df['PURCHASE_DATE'], errors='coerce')

# Convert FINAL_SALE to numeric
transactions_df['FINAL_SALE'] = pd.to_numeric(transactions_df['FINAL_SALE'], errors='coerce')
merged_data = pd.merge(transactions_df, users_df, left_on='USER_ID', right_on='ID', how='inner')
merged_data = pd.merge(merged_data, products_df, on='BARCODE', how='inner')
merged_data['BIRTH_YEAR'] = merged_data['BIRTH_DATE'].dt.year

# Define generations manually
generation_list = []
for year in merged_data['BIRTH_YEAR']:
    if pd.isna(year):
        generation_list.append('Unknown')
    elif year < 1946:
        generation_list.append('Silent Generation')
    elif year < 1965:
        generation_list.append('Baby Boomers')
    elif year < 1981:
        generation_list.append('Generation X')
    elif year < 1997:
        generation_list.append('Millennials')
    else:
        generation_list.append('Generation Z')


merged_data['GENERATION'] = generation_list
health data = merged data[merged data['CATEGORY 1'] == 'Health & Wellness']
```