Heroes Of Pymoli Data Analysis

- Of the 1163 active players, the vast majority are male (84%). There also exists, a smaller, but notable proportion of female players (14%).
- Our peak age demographic falls between 20-24 (44.8%) with secondary groups falling between 15-19 (18.60%) and 25-29 (13.4%).

Note

- Instructions have been included for each segment. You do not have to follow them exactly, but they are included to help you think through the steps.

In [1]:
```
# Dependencies and Setup
import pandas as pd
import numpy as np
from IPython.display import display, HTML
pd.options.display.float_format = '${:,.2f}'.format

# Raw data file
file_to_load = "Resources/purchase_data.csv"

# Read purchasing file and store into pandas data frame
purchase_data = pd.read_csv(file_to_load)
purchase_data.head()
```

Out[1]:

	Purchase ID	SN	Age	Gender	Item ID	Item Name	Price
0	0	Lisim78	20	Male	108	Extraction, Quickblade Of Trembling Hands	$3.53
1	1	Lisovynya38	40	Male	143	Frenzied Scimitar	$1.56
2	2	Ithergue48	24	Male	92	Final Critic	$4.88
3	3	Chamassasya86	24	Male	100	Blindscythe	$3.27
4	4	Iskosia90	23	Male	131	Fury	$1.44

Player Count

- Display the total number of players

In [2]:
```
#len(purchase_data["SN"].unique())
player_count = purchase_data["SN"].nunique()
pd.DataFrame({'Total Players':[player_count]})
```

Out[2]:

	Total Players
0	576

Purchasing Analysis (Total)

- Run basic calculations to obtain number of unique items, average price, etc.
- Create a summary data frame to hold the results
- Optional: give the displayed data cleaner formatting
- Display the summary data frame

In [3]:
```
# Unique item count
u_item_count = purchase_data["Item ID"].nunique()
u_item_count
```

Out[3]:
```
183
```

In [4]:
```
# Average price
avg_price = purchase_data["Price"].mean()
avg_price
round(avg_price,2)
```

Out[4]:
```
3.05
```

In [5]:
```
# Total number of purchases
total_purchases = purchase_data["Purchase ID"].nunique()
total_purchases
```

Out[5]:
```
780
```

In [6]:
```
# Total Revenue
total = purchase_data["Price"].sum()
total
```

Out[6]:
```
2379.77
```

In [7]:
```
pd.DataFrame({'Number of Unique Items' : [u_item_count],
              'Average Price' : [round(avg_price,2)],
              'Number of Purchases' : [total_purchases],
              'Total Revenue' : [total]})
```

Out[7]:

	Number of Unique Items	Average Price	Number of Purchases	Total Revenue
0	183	$3.05	780	$2,379.77

Gender Demographics

- Percentage and Count of Male Players
- Percentage and Count of Female Players
- Percentage and Count of Other / Non-Disclosed

In [8]:
```
# First way to compute count of players by gender
# df = purchase_data[purchase_data['Gender'] == 'Male']
# male_players = len(df.index)
# df = purchase_data[purchase_data['Gender'] == 'Female']
# female_players = len(df.index)
# df = purchase_data[(purchase_data['Gender'] != 'Female') & (purchase_data['G
ender'] != 'Male')]
# other_players = len(df.index)
# (male_players, female_players, other_players)
# Shape will print the total dimension of the dataframe and we extract number
 of rows
# df.shape[0]

purch_df = purchase_data.drop_duplicates(['SN', 'Gender'])
df = purch_df['Gender'].value_counts().to_frame()

# Alternative ways to get player count
# male_players = df[0]
#female_players = df[1]
#other_players = df[2]
# len(df.index)
# male_players, female_players, other_players = (df['Gender']/player_count) *
 100
# df['Percentage of Players']= male_players, female_players, other_players

# Percentage of players based on gender
players = (df['Gender'] * 100) / player_count
df['Percentage of Players']= players
df.columns
df.rename(columns={'Gender':'Total Count'}, inplace=True)
df = df [['Percentage of Players', 'Total Count']]
pd.options.display.float_format = '{:,.2f}'.format
df
```

Out[8]:

	Percentage of Players	Total Count
Male	84.03	484
Female	14.06	81
Other / Non-Disclosed	1.91	11

Purchasing Analysis (Gender)

- Run basic calculations to obtain purchase count, avg. purchase price, avg. purchase total per person etc. by gender
- Create a summary data frame to hold the results
- Optional: give the displayed data cleaner formatting
- Display the summary data frame

In [9]:
```
df = pd.DataFrame()
df_gpby = purchase_data.groupby(['Gender'])['Price']
df['Purchase Count'] = df_gpby.size()
df['Average Purchase Price'] = df_gpby.mean()
purch_sum = df_gpby.sum()
df['Total Purchase Value'] = purch_sum
ind_sum = purchase_data.groupby(['SN', 'Gender'])['Price'].sum()
ind_sum = ind_sum.reset_index()
df['Avg Total Purchase Value'] = ind_sum.groupby(['Gender'])['Price'].mean()
df
```

Out[9]:

	Purchase Count	Average Purchase Price	Total Purchase Value	Avg Total Purchase Value
Gender				
Female	113	3.20	361.94	4.47
Male	652	3.02	1,967.64	4.07
Other / Non-Disclosed	15	3.35	50.19	4.56

Age Demographics

- Establish bins for ages
- Categorize the existing players using the age bins. Hint: use pd.cut()
- Calculate the numbers and percentages by age group
- Create a summary data frame to hold the results
- Optional: round the percentage column to two decimal points
- Display Age Demographics Table

In [10]:
```
# Establish bins for ages
age_bins = [0, 9.90, 14.90, 19.90, 24.90, 29.90, 34.90, 39.90, 99999]
group_names = ["<10", "10-14", "15-19", "20-24", "25-29", "30-34", "35-39", "4
0+"]
pd.options.display.float_format = '{:,.2f}'.format
df_grpby = purchase_data.copy()
df_grpby.drop_duplicates(['SN', 'Gender'], inplace=True, keep='first')
df_grpby.head()
df_grpby['Age_bin'] = pd.cut(df_grpby['Age'], age_bins, labels=group_names)
df_grpby.head()
df = pd.DataFrame()
df['Percentage of Players']=(df_grpby.groupby(['Age_bin'])['Age'].count() * 10
0)/float(player_count)
df['Total Count'] = df_grpby.groupby(['Age_bin'])['Age'].count()
del df.index.name # Remove the groupby generated axis name
df
```

Out[10]:

	Percentage of Players	Total Count
<10	2.95	17
10-14	3.82	22
15-19	18.58	107
20-24	44.79	258
25-29	13.37	77
30-34	9.03	52
35-39	5.38	31
40+	2.08	12

Purchasing Analysis (Age)

- Bin the purchase_data data frame by age
- Run basic calculations to obtain purchase count, avg. purchase price, avg. purchase total per person etc. in the table below
- Create a summary data frame to hold the results
- Optional: give the displayed data cleaner formatting
- Display the summary data frame

In [11]:
```
# Establish bins for ages
age_bins = [0, 9.90, 14.90, 19.90, 24.90, 29.90, 34.90, 39.90, 99999]
group_names = ["<10", "10-14", "15-19", "20-24", "25-29", "30-34", "35-39", "4
0+"]
pd.options.display.float_format = '${:,.2f}'.format

#  To create a copy. Done only for practice. Can be done without copy
df_grpby = purchase_data.copy()

#  Bin the values and create a new column
df_grpby['Age_bin'] = pd.cut(df_grpby['Age'], age_bins, labels=group_names)
df_grpby.head()

#  Create new empty data frame
df = pd.DataFrame()
df['Purchase Count']= df_grpby.groupby(['Age_bin'])['Purchase ID'].count()
df['Average Purchase Price'] = df_grpby.groupby(['Age_bin'])['Price'].mean()
df['Total Purchase Value'] = df_grpby.groupby(['Age_bin'])['Price'].sum()
del df.index.name
df_1 = df_grpby.groupby(['Age_bin', 'SN'])['Price'].sum()
#  Compute Avg purchase total per person. The values are slightly different
#  than shown in sample answer as sample answer does not groupby SN column
#  Confirmed from Tyler that my computation is correct as per the stated quest
ion
#  df_2 = df_grpby.groupby(['Age_bin', 'SN'])['Price'].count()
#  df_3 = df_1/df_2
#  df_3

df_1 = df_1.reset_index()
df['Average Purchase Total per Person'] = df_1.groupby(['Age_bin'])['Price'].m
ean()
df
```

Out[11]:

	Purchase Count	Average Purchase Price	Total Purchase Value	Average Purchase Total per Person
<10	23	$3.35	$77.13	$4.54
10-14	28	$2.96	$82.78	$3.76
15-19	136	$3.04	$412.89	$3.86
20-24	365	$3.05	$1,114.06	$4.32
25-29	101	$2.90	$293.00	$3.81
30-34	73	$2.93	$214.00	$4.12
35-39	41	$3.60	$147.67	$4.76
40+	13	$2.94	$38.24	$3.19

Top Spenders

- Run basic calculations to obtain the results in the table below
- Create a summary data frame to hold the results
- Sort the total purchase value column in descending order
- Optional: give the displayed data cleaner formatting
- Display a preview of the summary data frame

In [12]:
```
pd.options.display.float_format = '${:,.2f}'.format
df = purchase_data.copy()
summary_df = pd.DataFrame()
df.head()
df.set_index('SN', inplace=True)
summary_df['Purchase Count'] = df.groupby(['SN'])['Item ID'].count()
summary_df['Average Purchase Price'] = df.groupby(['SN'])['Price'].mean()
summary_df['Total Purchase Value'] = df.groupby(['SN'])['Price'].sum()
summary_df.sort_values('Total Purchase Value', ascending=False).head()
```

Out[12]:

	Purchase Count	Average Purchase Price	Total Purchase Value
SN			
Lisosia93	5	$3.79	$18.96
Idastidru52	4	$3.86	$15.45
Chamjask73	3	$4.61	$13.83
Iral74	4	$3.40	$13.62
Iskadarya95	3	$4.37	$13.10

Most Popular Items

- Retrieve the Item ID, Item Name, and Item Price columns
- Group by Item ID and Item Name. Perform calculations to obtain purchase count, item price, and total purchase value
- Create a summary data frame to hold the results
- Sort the purchase count column in descending order
- Optional: give the displayed data cleaner formatting
- Display a preview of the summary data frame

```
In [13]: df = purchase_data.copy()
         summary_df = pd.DataFrame()
         df = df[['Item ID', 'Item Name', 'Price']]
         summary_df['Purchase Count'] = df.groupby(['Item ID', 'Item Name'])['Item ID']
         .count()
         df_sum = df.groupby(['Item ID', 'Item Name'])['Price'].sum()
         df_count = df.groupby(['Item ID', 'Item Name'])['Price'].count()
         summary_df['Item Price'] = df_sum/df_count
         summary_df['Total Purchase Value'] = df_sum
         #df_grp.sort_values('Item ID', ascending=False)
         summary_df.sort_values("Purchase Count", ascending=False).head(5)
```

Out[13]:

		Purchase Count	Item Price	Total Purchase Value
Item ID	**Item Name**			
178	**Oathbreaker, Last Hope of the Breaking Storm**	12	$4.23	$50.76
145	**Fiery Glass Crusader**	9	$4.58	$41.22
108	**Extraction, Quickblade Of Trembling Hands**	9	$3.53	$31.77
82	**Nirvana**	9	$4.90	$44.10
19	**Pursuit, Cudgel of Necromancy**	8	$1.02	$8.16

Most Profitable Items

- Sort the above table by total purchase value in descending order
- Optional: give the displayed data cleaner formatting
- Display a preview of the data frame

In [14]:
```
summary_df.sort_values("Total Purchase Value", ascending=False).head(5)
```

Out[14]:

		Purchase Count	Item Price	Total Purchase Value
Item ID	**Item Name**			
178	**Oathbreaker, Last Hope of the Breaking Storm**	12	$4.23	$50.76
82	**Nirvana**	9	$4.90	$44.10
145	**Fiery Glass Crusader**	9	$4.58	$41.22
92	**Final Critic**	8	$4.88	$39.04
103	**Singed Scalpel**	8	$4.35	$34.80