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 [29]:
```
# Dependencies and Setup
import pandas as pd
import numpy as np

# File to Load (Remember to Change These)
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[29]:

	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 [43]:
```
#find unique players
players = len(purchase_data['SN'].value_counts())
#show number of unique players
unique_players = pd.DataFrame({"Total Players": [players]})
unique_players
```

Out[43]:

	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 [44]:
```
#total purchases
purchases = purchase_data["Purchase ID"].count()
#unique items
unique_items = len(purchase_data["Item ID"].unique())
#average price
avg_p = purchase_data["Price"].mean()
#total revenue
total_rev = purchase_data["Price"].sum()
print(purchases)
print(unique_items)
print(avg_p)
print(total_rev)
```

```
780
183
3.050987179487176
2379.77
```

In [45]:
```
#Purchasing Analysis DataFrame
summary = [("Number of Unique Items", [unique_items]),
           ("Average Price", [avg_p]),
           ("Number of Purchases", [purchases]),
           ("Total Revenue", [total_rev])]
df_summary = pd.DataFrame.from_items(summary)
#Editing the formatting
df_summary["Average Price"]=df_summary["Average Price"].map("${:,.2f}".format)
df_summary["Total Revenue"]=df_summary["Total Revenue"].map("${:,.2f}".format)
df_summary
```

```
C:\Users\jerri\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: FutureWarning: from_items is deprecated.
Please use DataFrame.from_dict(dict(items), ...) instead. DataFrame.from_dict(OrderedDict(items)) may be used
to preserve the key order.
```

Out[45]:

	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 [48]:
```
#Find total gender, and total of m, f, non
duplicate = purchase_data.drop_duplicates(subset='SN', keep="first")
gender_total = duplicate["Gender"].count()
m_total = duplicate["Gender"].value_counts()['Male']
f_total = duplicate["Gender"].value_counts()['Female']
non_total = TotalGen - MaleGen - FemaleGen

m_per = (m_total / gender_total) * 100
f_per = (f_total / gender_total) * 100
non_per = (non_total / gender_total) * 100

# Create new DataFrame
gender_df = pd.DataFrame({"": ['Male', 'Female', 'Other/Non-Disclosed'],
                          "Percentage of Players": [m_per, f_per, non_per],
                          "Total Count": [m_total, f_total, non_total]})

# DataFrame formatting
gender_df["Percentage of Players"] = gender_df["Percentage of Players"].map("{:.2f}%".format)
gender_df
```

Out[48]:

		Percentage of Players	Total Count
0	Male	84.03%	484
1	Female	14.06%	81
2	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 [58]:
```
#group by Gender
grouped_df = purchase_data.groupby(["Gender"])

#count, avg, total
purch_count = grouped_df["SN"].count()
purch_avg = grouped_df["Price"].mean()
purch_total = grouped_df["Price"].sum()

#removing duplicates
duplicate_grouped_df = duplicate.groupby(["Gender"])
duplicate_purch = (grouped_df["Price"].sum() / duplicate_grouped_df["SN"].count())

#Creating new dataframe
purch_analysis = pd.DataFrame({"Purchase Count": purch_count,
                               "Average Purchase Price": purch_avg,
                               "Total Purchase Value": purch_total,
                               "Avg Total Purchases per Person": duplicate_purch})

#formatting
purch_analysis["Average Purchase Price"] = purch_analysis["Average Purchase Price"].map("${:.2f}".format)
purch_analysis["Total Purchase Value"] = purch_analysis["Total Purchase Value"].map("${:,.2f}".format)
purch_analysis["Avg Total Purchases per Person"] = purch_analysis["Avg Total Purchases per Person"].map("${:.
2f}".format)
purch_analysis = purch_analysis[["Purchase Count", "Average Purchase Price", "Total Purchase Value", "Avg Tot
al Purchases per Person"]]
purch_analysis
```

Out[58]:

Gender	Purchase Count	Average Purchase Price	Total Purchase Value	Avg Total Purchases per Person
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 [67]:
```
#set up bins
bins = [0,9.99,14.99,19.99,24.99,29.99,34.99,39.99,2000]
names = ['<10', '10-14', '15-19', '20-24', '25-29', '30-34', '35-39', '40+']

#add bins to dataframe
bins_df = purchase_data.copy()
bins_df["Age Groups"] = pd.cut(bins_df["Age"], bins, labels=names)
bins_group = bins_df.groupby(["Age Groups"])

#find counts and %
total_count = purchase_data["SN"].nunique()
bins_count = bins_group["SN"].nunique()
bins_pct = (bins_count / total_count) * 100
bins_pct

#dataframe
age_df = pd.DataFrame({"Total Count": bins_count, "Percentage of Players": bins_pct})

#formatting
age_df["Percentage of Players"] = age_df["Percentage of Players"].map("{:.2f}%".format)
age_df
```

Out[67]:

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

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 [70]:
```
#set up bins
bins = [0,9.99,14.99,19.99,24.99,29.99,34.99,39.99,2000]
names = ['<10', '10-14', '15-19', '20-24', '25-29', '30-34', '35-39', '40+']

#add bins to dataframe
bins_df = purchase_data.copy()
bins_df["Age Groups"] = pd.cut(bins_df["Age"], bins, labels=names)
bins_column = pd.cut(bins_df["Age"], bins, labels=names)
bins_group = bins_df.groupby(["Age Groups"])

#find counts, avgs, totals
bins_counts = bins_group["Purchase ID"].count()
bins_avgs = bins_group["Price"].mean()
bins_totals = bins_group["Price"].sum()

#calculate avg purchase per person
avg_purch = bins_totals/total_count

#dataframe
purch_age_analysis = pd.DataFrame({"Purchase Count": bins_counts,
                                   "Average Purchase Price": bins_avgs,
                                   "Total Purchase Value": bins_totals,
                                   "Average Purchase Total per Person": avg_purch})
#formatting
purch_age_analysis.style.format({"Average Purchase Price":"${:,.2f}",
                                 "Total Purchase Value":"${:,.2f}",
                                 "Average Purchase Total per Person":"${:,.2f}"})
```

Out[70]:

Age Groups	Purchase Count	Average Purchase Price	Total Purchase Value	Average Purchase Total per Person
<10	23	$3.35	$77.13	$0.13
10-14	28	$2.96	$82.78	$0.14
15-19	136	$3.04	$412.89	$0.72
20-24	365	$3.05	$1,114.06	$1.93
25-29	101	$2.90	$293.00	$0.51
30-34	73	$2.93	$214.00	$0.37
35-39	41	$3.60	$147.67	$0.26
40+	13	$2.94	$38.24	$0.07

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 [77]:
```
#group by SN
spender_df = purchase_data.groupby("SN")

#count, avg and total
spender_count = spender_df["Purchase ID"].count()
spender_avg = spender_df["Price"].mean()
spender_total = spender_df["Price"].sum()

#dataframe
top_spenders = pd.DataFrame({"Purchase Count": spender_count,
                             "Average Purchase Price": spender_avg,
                             "Total Purchase Value": spender_total})

#descending order
top_spender_df = top_spenders.sort_values(["Total Purchase Value"], ascending=False).head(5)

#formatting
top_spender_df.style.format({"Average Purchase Total":"${:,.2f}",
                             "Average Purchase Price":"${:,.2f}",
                             "Total Purchase Value":"${:,.2f}"})
```

Out[77]:

SN	Purchase Count	Average Purchase Price	Total Purchase Value
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 [80]:
```
#items dataframe and group
items = purchase_data[["Item ID", "Item Name", "Price"]]
item_stats = items.groupby(["Item ID", "Item Name"])

#count of purchases, value of item and individual price of item
item_count = item_stats["Price"].count()
item_total = (item_stats["Price"].sum())
item_price = item_total/item_count

#dataframe
most_popular = pd.DataFrame({"Purchase Count": item_count,
                             "Item Price": item_price,
                             "Total Purchase Value":item_total})

#decsending order & formatting
most_popular_df = most_popular.sort_values(["Purchase Count"], ascending=False).head(5)
most_popular_df.style.format({"Item Price":"${:,.2f}",
                              "Total Purchase Value":"${:,.2f}"})
```

Out[80]:

		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 [83]:
```
#sort previous table on total purchase value
most_popular_df = most_popular.sort_values(["Total Purchase Value"], ascending=False).head(5)
most_popular_df.style.format({"Item Price":"${:,.2f}",
                              "Total Purchase Value":"${:,.2f}"})
```

Out[83]:

Item ID	Item Name	Purchase Count	Item Price	Total Purchase Value
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

In [ ]:
```

```