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 [52]:
```
# Dependencies and Setup
import pandas as pd
import numpy as np

# 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[52]:

	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 [31]:
```
players=purchase_data["SN"].unique()
total_players=len(players)
print("total players:"+str(total_players))
```

```
total players: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 [54]:
```
items=purchase_data["Item ID"].unique()
total_items=len(items)
avg_price=purchase_data["Price"].mean()
purchases=purchase_data["Purchase ID"].value_counts()
purchases_count=len(purchases)
total_revenue=purchase_data["Price"].sum()

summary_items={"Number of Unique Items":[total_items],
          "Average Price":[avg_price],
          "Number of Purchases":[purchases_count],
          "Total Revenue":[total_revenue]
}
df_items=pd.DataFrame(summary_items)
df_items["Average Price"] = df_items["Average Price"].map("${:.2f}".format)
df_items["Total Revenue"] = df_items["Total Revenue"].map("${:.2f}".format)
df_items
```

Out[54]:

	Number of Unique Items	Average Price	Number of Purchases	Total Revenue
0	183	$3.05	780	$2379.77

Gender Demographics

```
* 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 [196]:
```
gender_count=purchase_data["Gender"].value_counts().tolist()
gender=["Male","Female","Others/Non-disclosed"]
gender_percentage=[]
for i in range(len(gender_count)):
    gender_percentage.append(gender_count[i]/total_players)

summary_gender={"":gender,
                "Total Count":gender_count,
                "Percentage of Players":gender_percentage
}
df_gender=pd.DataFrame.from_dict(summary_gender)
df_gender["Percentage of Players"] = df_gender["Percentage of Players"].map(
df_gender
```

Out[196]:

		Total Count	Percentage of Players
0	Male	652	1.13%
1	Female	113	0.20%
2	Others/Non-disclosed	15	0.03%

Purchasing Analysis (Gender)

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

In [203]:
```
grouped_bygender=purchase_data.groupby(["Gender"])
avg_price=grouped_bygender["Price"].mean()
total_purchase=grouped_bygender["Price"].sum()
normalised=grouped_bygender["Price"].mean()
summary_gender_purchase={"Purchase Count":[113,652,15],
                         "Average Purchase Price":avg_price,
                         "Total Purchase Value":total_purchase,
                         "Normalized Totals":normalised
                         }
df_grouped_bygender=pd.DataFrame(summary_gender_purchase)
df_grouped_bygender["Average Purchase Price"] = df_grouped_bygender["Average
df_grouped_bygender["Total Purchase Value"] = df_grouped_bygender["Total Pur
df_grouped_bygender["Normalized Totals"] =df_grouped_bygender["Normalized To
df_grouped_bygender
```

Out[203]:

Gender	Purchase Count	Average Purchase Price	Total Purchase Value	Normalized Totals
Female	113	$3.20	$361.94	$3.20
Male	652	$3.02	$1967.64	$3.02
Other / Non-Disclosed	15	$3.35	$50.19	$3.35

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 [110]:
```
# 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",
purchase_data["age_bin"] = pd.cut(purchase_data["Age"], age_bins, labels=gro
grouped_by_bins=purchase_data.groupby(["age_bin"])
total_count_byage=grouped_by_bins["SN"].size()
percentage_byage=[]
for k in range(len(total_count_byage)):
    percentage_byage.append((total_count_byage[k]/total_players)*100)
summary_bybins={"Percentage of Players":percentage_byage,
                        "Total Count":total_count_byage,
                        }
df_bin=pd.DataFrame(summary_bybins)
df_bin["Percentage of Players"] = df_bin["Percentage of Players"].map("{:.2f
df_bin
```

Out[110]:

age_bin	Percentage of Players	Total Count
<10	3.99	23
10-14	4.86	28
15-19	23.61	136
20-24	63.37	365
25-29	17.53	101
30-34	12.67	73
35-39	7.12	41
40+	2.26	13

Purchasing Analysis (Age)

* Bin the purchase_data data frame by age

* Run basic calculations to obtain purchase count, avg. purchase price, etc. in the table below

* Calculate Normalized Purchasing

* Create a summary data frame to hold the results

* Optional: give the displayed data cleaner formatting

* Display the summary data frame

In [116]:
```
avg_purchase_bybin=grouped_by_bins["Price"].mean()
total_purchase_bybin=grouped_by_bins["Price"].sum()
summary_purchasebybins={"Purchase Count":total_count_byage,
                "Average Purchase Price":avg_purchase_bybin,
                "Total Purchase Value":total_purchase_bybin,
                "Normalized Totals":avg_purchase_bybin
                        }
df_purchasebin=pd.DataFrame(summary_purchasebybins)
df_purchasebin["Average Purchase Price"] = df_purchasebin["Average Purchase
df_purchasebin["Total Purchase Value"] = df_purchasebin["Total Purchase Valu
df_purchasebin["Normalized Totals"] = df_purchasebin["Normalized Totals"].ma
df_purchasebin
```

Out[116]:

age_bin	Purchase Count	Average Purchase Price	Total Purchase Value	Normalized Totals
<10	23	$3.35	$77.13	$3.35
10-14	28	$2.96	$82.78	$2.96
15-19	136	$3.04	$412.89	$3.04
20-24	365	$3.05	$1114.06	$3.05
25-29	101	$2.90	$293.00	$2.90
30-34	73	$2.93	$214.00	$2.93
35-39	41	$3.60	$147.67	$3.60
40+	13	$2.94	$38.24	$2.94

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 [129]:
```
grouped_bySN=purchase_data.groupby(["SN"])
SN_purchasecount=grouped_bySN["Purchase ID"].size()
SN_averageprice=grouped_bySN["Price"].mean()
SN_Totalprice=grouped_bySN["Price"].sum()
SN_Totalprice
summary_SN_purchase={"Purchase Count":SN_purchasecount,
                     "Average Purchase Price":SN_averageprice,
                     "Total Purchase Value":SN_Totalprice
                     }
df_grouped_bySN=pd.DataFrame(summary_SN_purchase)
sorted=df_grouped_bySN.sort_values(by='Total Purchase Value', ascending=Fals
df_grouped_bySN["Average Purchase Price"] = df_grouped_bySN["Average Purchas
df_grouped_bySN["Total Purchase Value"] = df_grouped_bySN["Total Purchase Va
sorted.head()
```

Out[129]:

SN	Purchase Count	Average Purchase Price	Total Purchase Value
Lisosia93	5	3.792000	18.96
Idastidru52	4	3.862500	15.45
Chamjask73	3	4.610000	13.83
Iral74	4	3.405000	13.62
Iskadarya95	3	4.366667	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 [160]:
```
grouped_byItem=purchase_data.groupby(["Item ID","Item Name"])
Item_purchasecount=grouped_byItem["Purchase ID"].size()
Item_totalpurchase=grouped_byItem["Price"].sum()
Item_price=[]
for i in range(len(Item_purchasecount)):
    Item_price.append(Item_totalpurchase.tolist()[i]/Item_purchasecount.toli
summary_item_purchase={"Purchase Count":Item_purchasecount,
                       "Item Price":Item_price,
                       "Total Purchase Value":Item_totalpurchase
                       }
df_grouped_byItem=pd.DataFrame(summary_item_purchase)
sorted_item_pop=df_grouped_byItem.sort_values(by='Purchase Count', ascending
sorted_item_pop["Item Price"] = sorted_item_pop["Item Price"].map("${:.2f}"
sorted_item_pop["Total Purchase Value"] = sorted_item_pop["Total Purchase Va
sorted_item_pop.head()
```

Out[160]:

Item ID	Item Name	Purchase Count	Item Price	Total Purchase Value
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 [161]:
```
sorted_item_prof=df_grouped_byItem.sort_values(by='Total Purchase Value', as
sorted_item_prof["Item Price"] = sorted_item_prof["Item Price"].map("${:.2f}
sorted_item_prof["Total Purchase Value"] = sorted_item_prof["Total Purchase
sorted_item_prof.head()
```

Out[161]:

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 [ ]:
```

```