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

# 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[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]:

```
# Total players - find unqiue number of players
unique_players = len(purchase_data["SN"].value_counts())
total_players = {"Total # of unqiue players that made purchases": [unique_play
ers]}
total_players_df = pd.DataFrame(total_players)
total_players_df
```

Out[2]:

	Total # of unqiue players that made purchases
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]:

```
# Number of Unqiue Items
num_unq = len(purchase_data["Item ID"].value_counts())

# Average Price
avg_price = (purchase_data["Price"].mean())

# Number of Purchases
num_purch = (purchase_data["Price"].count())

# Total Revenue
total_rev = (purchase_data["Price"].sum())

# Create dataframe
purch_overview = {"Number of Unique Items": [num_unq], "Average Price":[avg_pr
ice], "Number of Purchases":[num_purch], "Total Revenue":[total_rev]}
purch_overview_df = pd.DataFrame(purch_overview)

# Format dataframe columns
purch_overview_df["Average Price"] = purch_overview_df["Average Price"].map("$
{:.2f}".format)
purch_overview_df["Total Revenue"] = purch_overview_df["Total Revenue"].map("$
{:.2f}".format)

purch_overview_df
```

Out[3]:

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

Gender Demographics

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

In [4]:

```python
gender_data = purchase_data.groupby(["Gender"])
gender_data.count()

# Total Count by gender
male = len(purchase_data.loc[purchase_data["Gender"]=="Male", :])
female = len(purchase_data.loc[purchase_data["Gender"]=="Female", :])
non_dis = len(purchase_data.loc[purchase_data["Gender"]=="Other / Non-Disclosed", :])


# Percent of Players by gender
percent_male = (male/(male + female + non_dis)) * 100
percent_female = (female/(male + female + non_dis)) * 100
percent_non = (non_dis/(male + female + non_dis)) * 100


# Create dataframe
gender_data_df = pd.DataFrame(
[[percent_female, female],
[percent_male, male],
[percent_non, non_dis]],
index=["Female","Male", "Other / Non-Disclosed"],
columns=["Percentage of Players", "Total Count"])

# Format dataframe columns
gender_data_df["Percentage of Players"] = gender_data_df["Percentage of Players"].map("{:.2f}%".format)
gender_data_df
```

Out[4]:

	Percentage of Players	Total Count
Female	14.49%	113
Male	83.59%	652
Other / Non-Disclosed	1.92%	15

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 [5]:

```
# Grouby from previous code block
gender_data.count()

# Purchase COUNT calculated in previous code block -- variables are: "female", "male", and "non_dis" and used below.

# Average Purchase Price
female_avg_price = purchase_data.loc[purchase_data["Gender"]=="Female", "Price"].mean()
male_avg_price = purchase_data.loc[purchase_data["Gender"]=="Male", "Price"].mean()
non_avg_price = purchase_data.loc[purchase_data["Gender"]=="Other / Non-Disclosed", "Price"].mean()

# Total Purchase Value
female_tot_val = purchase_data.loc[purchase_data["Gender"]=="Female", "Price"].sum()
male_tot_val = purchase_data.loc[purchase_data["Gender"]=="Male", "Price"].sum()
non_tot_val = purchase_data.loc[purchase_data["Gender"]=="Other / Non-Disclosed", "Price"].sum()

# Average Purchase Total Per
female_avg_tot_per = female_tot_val / female
male_avg_tot_per = male_tot_val / male
non_avg_tot_per = non_tot_val / non_dis

# Create dataframe
per_gender_data_df = pd.DataFrame(
[[female, female_avg_price, female_tot_val, female_avg_tot_per],
[male, male_avg_price, male_tot_val, male_avg_tot_per],
[non_dis, non_avg_price, non_tot_val, non_avg_tot_per]],
index=["Female","Male", "Other / Non-Disclosed"],
columns=["Purchase Count", "Average Purchase Price", "Total Purchase Value", "Avg Purchase Total Per Person"])

# Format dataframe columns
per_gender_data_df["Average Purchase Price"] = per_gender_data_df["Average Purchase Price"].map("${:.2f}".format)
per_gender_data_df["Total Purchase Value"] = per_gender_data_df["Total Purchase Value"].map("${:.2f}".format)
per_gender_data_df["Avg Purchase Total Per Person"] = per_gender_data_df["Avg Purchase Total Per Person"].map("${:.2f}".format)

per_gender_data_df
```

Out[5]:

	Purchase Count	Average Purchase Price	Total Purchase Value	Avg Purchase Total Per Person
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 [6]:

```python
# 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", "40+"]

purchase_data["Age"] = pd.cut(purchase_data["Age"], age_bins, labels=group_names)

# Group by age column
age_data = purchase_data.groupby(["Age"])
age_data.count()

# Total Count by age group
age_u10 = len(purchase_data.loc[purchase_data["Age"]=="<10", :])
age_10_14 = len(purchase_data.loc[purchase_data["Age"]=="10-14", :])
age_15_19 = len(purchase_data.loc[purchase_data["Age"]=="15-19", :])
age_20_24 = len(purchase_data.loc[purchase_data["Age"]=="20-24", :])
age_25_29 = len(purchase_data.loc[purchase_data["Age"]=="25-29", :])
age_30_34 = len(purchase_data.loc[purchase_data["Age"]=="30-34", :])
age_35_39 = len(purchase_data.loc[purchase_data["Age"]=="35-39", :])
age_40 = len(purchase_data.loc[purchase_data["Age"]=="40+", :])

# Create variable for total
total = age_u10 + age_10_14 + age_15_19 + age_20_24 + age_25_29 + age_30_34 + age_35_39 + age_40

# Percent of Players by age group
percent_u10 = (age_u10/total) * 100
percent_10_14 = (age_10_14/total) * 100
percent_15_19 = (age_15_19/total) * 100
percent_20_24 = (age_20_24/total) * 100
percent_25_29 = (age_25_29/total) * 100
percent_30_34 = (age_30_34/total) * 100
percent_35_39 = (age_35_39/total) * 100
percent_age_40 = (age_40/total) * 100

# Create dataframe
age_data_df = pd.DataFrame(
[[percent_u10, age_u10],
[percent_10_14, age_10_14],
[percent_15_19, age_15_19],
[percent_20_24, age_20_24],
[percent_25_29, age_25_29],
[percent_30_34, age_30_34],
[percent_35_39, age_35_39],
[percent_age_40, age_40]],
index=["<10","10-14", "15-19", "20-24", "25-29", "30-34", "35-39", "40+"],
columns=["Percentage of Players", "Total Count"])

# Format dataframe columns
age_data_df["Percentage of Players"] = age_data_df["Percentage of Players"].map("{:.2f}%".format)
age_data_df
```

Out[6]:

	Percentage of Players	Total Count
<10	2.95%	23
10-14	3.59%	28
15-19	17.44%	136
20-24	46.79%	365
25-29	12.95%	101
30-34	9.36%	73
35-39	5.26%	41
40+	1.67%	13

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 [7]:

```
# Bins and groupby from previous code block

age_data.count()

# purchase COUNT calculated in previous code block -- variables are: "age_u1
0", "age_15_19",
# "age_20_24", "age_25_29", "age_30_34", "age_35_39", "age_40" and used below.

# Average Purchase Price
u10_avg_price = purchase_data.loc[purchase_data["Age"]=="<10", "Price"].mean()
age1014_avg_price = purchase_data.loc[purchase_data["Age"]=="10-14", "Price"].
mean()
age1519_avg_price = purchase_data.loc[purchase_data["Age"]=="15-19", "Price"].
mean()
age2024_avg_price = purchase_data.loc[purchase_data["Age"]=="20-24", "Price"].
mean()
age2529_avg_price = purchase_data.loc[purchase_data["Age"]=="25-29", "Price"].
mean()
age3034_avg_price = purchase_data.loc[purchase_data["Age"]=="30-34", "Price"].
mean()
age3539_avg_price = purchase_data.loc[purchase_data["Age"]=="35-39", "Price"].
mean()
age40_avg_price = purchase_data.loc[purchase_data["Age"]=="40+", "Price"].mean
()

# Total Purchase Value
u10_tot_val = purchase_data.loc[purchase_data["Age"]=="<10", "Price"].sum()
# print(u10_tot_val)
age1014_tot_val = purchase_data.loc[purchase_data["Age"]=="10-14", "Price"].su
m()
# print(age1014_tot_val)
age1519_tot_val = purchase_data.loc[purchase_data["Age"]=="15-19", "Price"].su
m()
# print(age1519_tot_val)
age2024_tot_val = purchase_data.loc[purchase_data["Age"]=="20-24", "Price"].su
m()
# print(age2024_tot_val)
age2529_tot_val = purchase_data.loc[purchase_data["Age"]=="25-29", "Price"].su
m()
# print(age2529_tot_val)
age3034_tot_val = purchase_data.loc[purchase_data["Age"]=="30-34", "Price"].su
m()
# print(age3034_tot_val)
age3539_tot_val = purchase_data.loc[purchase_data["Age"]=="35-39", "Price"].su
m()
# print(age3539_tot_val)
age40_tot_val = purchase_data.loc[purchase_data["Age"]=="40+", "Price"].sum()
# print(age40_tot_val)

# Average Purchase Total Per
u10_avg_tot_per = u10_tot_val / age_u10
# print(u10_avg_tot_per)
age1014_avg_tot_per = age1014_tot_val / age_10_14
# print(male_avg_tot_per)
age1519_avg_tot_per = age1519_tot_val / age_15_19
# print(age1519_avg_tot_per)
```

```
age2024_avg_tot_per = age2024_tot_val / age_20_24
# print(age2024_avg_tot_per)
age2529_avg_tot_per = age2529_tot_val / age_25_29
# print(age2529_avg_tot_per)
age3034_avg_tot_per = age3034_tot_val / age_30_34
# print(age3034_avg_tot_per)
age3539_avg_tot_per = age3539_tot_val / age_35_39
# print(age3539_avg_tot_per)
age40_avg_tot_per = age40_tot_val / age_40
# print(age40_avg_tot_per)

# Create dataframe
per_age_data_df = pd.DataFrame(
[[age_u10, u10_avg_price, u10_tot_val, u10_avg_tot_per],
[age_10_14, age1014_avg_price, age1014_tot_val, age1014_avg_tot_per],
[age_15_19, age1519_avg_price, age1519_tot_val, age1519_avg_tot_per],
[age_20_24, age2024_avg_price, age2024_tot_val, age2024_avg_tot_per],
[age_25_29, age2529_avg_price, age2529_tot_val, age2529_avg_tot_per],
[age_30_34, age3034_avg_price, age3034_tot_val, age3034_avg_tot_per],
[age_35_39, age3539_avg_price, age3539_tot_val, age3539_avg_tot_per],
[age_40, age40_avg_price, age40_tot_val, age40_avg_tot_per]],
index=["<10","10-14", "15-19", "20-24", "25-29", "30-34", "35-39", "40+"],
columns=["Purchase Count", "Average Purchase Price", "Total Purchase Value",
"Avg Purchase Total Per Person"])

# Formate dataframe columns
per_age_data_df["Average Purchase Price"] = per_age_data_df["Average Purchase
 Price"].map("${:.2f}".format)
per_age_data_df["Total Purchase Value"] = per_age_data_df["Total Purchase Valu
e"].map("${:.2f}".format)
per_age_data_df["Avg Purchase Total Per Person"] = per_age_data_df["Avg Purcha
se Total Per Person"].map("${:.2f}".format)

per_age_data_df
```

Out[7]:

	Purchase Count	Average Purchase Price	Total Purchase Value	Avg Purchase Total Per Person
<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 [8]:

```
# Trying to use less code!

total_purchase = purchase_data.groupby("SN")["Price"].sum()
purchase_count = purchase_data.groupby("SN")["Price"].count()
purchase_avg = purchase_data.groupby("SN")["Price"].mean()

# Create DataFrame
spender_df = pd.DataFrame({"Purchase Count": purchase_count, "Average Purchase
Price": purchase_avg, "Total Purchase Value": total_purchase})

# Sort by Total Purchase Value
spender_df.sort_values("Total Purchase Value", ascending = False, inplace=True
)

# Format columns
spender_df["Average Purchase Price"] = spender_df["Average Purchase Price"].ma
p("${:.2f}".format)
spender_df["Total Purchase Value"] = spender_df["Total Purchase Value"].map("$
{:.2f}".format)

# Show Top 5 only
spender_df.head()
```

Out[8]:

	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 [9]:

```
pop_tot = purchase_data.groupby("Item Name")["Price"].sum()
pop_count = purchase_data.groupby("Item Name")["Price"].count()
# Used .mean on item price --- there was a variance on item "Final Critic" (max price = 4.88, min price = 4.19)
pop_price = purchase_data.groupby("Item Name")["Price"].mean()

# Create DataFrame
pop_df = pd.DataFrame({"Purchase Count": pop_count, "Avg Item Price": pop_price, "Total Purchase Value": pop_tot})

# Sort by Purchase Count
pop_df.sort_values("Purchase Count", ascending = False, inplace=True)

# Format columns
pop_df["Avg Item Price"] = pop_df["Avg Item Price"].map("${:.2f}".format)
pop_df["Total Purchase Value"] = pop_df["Total Purchase Value"].map("${:.2f}".format)

# Show Top 5 only
pop_df.head()
```

Out[9]:

	Purchase Count	Avg Item Price	Total Purchase Value
Item Name			
Final Critic	13	$4.61	$59.99
Oathbreaker, Last Hope of the Breaking Storm	12	$4.23	$50.76
Persuasion	9	$3.22	$28.99
Nirvana	9	$4.90	$44.10
Extraction, Quickblade Of Trembling Hands	9	$3.53	$31.77

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 [10]:
```
# Create new DataFrame name using variables from previous code block
prof_df = pd.DataFrame({"Purchase Count": pop_count, "Avg Item Price": pop_price, "Total Purchase Value": pop_tot})

# Sort by Total Purchase Value
prof_df.sort_values("Total Purchase Value", ascending = False, inplace=True)

# Format columns
prof_df["Avg Item Price"] = prof_df["Avg Item Price"].map("${:.2f}".format)
prof_df["Total Purchase Value"] = prof_df["Total Purchase Value"].map("${:.2f}".format)

# Show Top 5 only
prof_df.head()
```

Out[10]:

	Purchase Count	Avg Item Price	Total Purchase Value
Item Name			
Final Critic	13	$4.61	$59.99
Oathbreaker, Last Hope of the Breaking Storm	12	$4.23	$50.76
Nirvana	9	$4.90	$44.10
Fiery Glass Crusader	9	$4.58	$41.22
Singed Scalpel	8	$4.35	$34.80