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 [65]:
```
# Dependencies and Setup
import pandas as pd
import numpy as np

# File to Load (Remember to Change These)
#purchase_data.csv = "Resources/purchase_data.csv"

# Read Purchasing File and store into Pandas data frame
# import a CSV file and rename csv for ease of use
data = pd.read_csv("purchase_data.csv")
```

Player Count

- Display the total number of players
- Cleaner formatting Player Count

In [66]:
```
SN = {'Total Player':[len(data['SN'].value_counts())]}
SN1 = pd.DataFrame(SN)
SN1
```

Out[66]:

	Total Player
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 [67]:
```
it = (data['Item ID'].nunique())
print(it)
```

```
183
```

** Average price

In [68]:
```
average_price = (data['Price'].mean())
print(average_price)
```

```
3.050987179487176
```

In [69]:
```
purchased_number = (data['SN'].count())
print(purchased_number)
```

```
780
```

In [70]:
```
total_revenue = (data['Price'].sum())
print(total_revenue)
```

```
2379.77
```

** Cleaner formatting Player Count

In [71]:
```
#Number of Unique Items
UniqueItems = len(data['Item Name'].value_counts())
#Average Purchase Price
AvPurchPrice = round(data['Price'].mean(),2)
#Total Purchase Value
TotalPurchases = round(data['Price'].sum(),2)
#Purchasing Analysis (Total)
PurchasingAnalysis= {'Unique Items':[UniqueItems],'Average Price':[AvPurchPr
PurchasingAnalysis1 =pd.DataFrame(PurchasingAnalysis)
PurchasingAnalysis1 = PurchasingAnalysis1[['Unique Items','Average Price','N
PurchasingAnalysis1
```

Out[71]:

	Unique Items	Average Price	Number of Purchases	Total Purchases
0	179	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 [72]:

```
total_count = len(data["SN"].unique())

#Percentage and Count of Male Players
#Male = data.loc[data['Gender'] =='Male',:]
Male1= data.groupby(['Gender']).get_group(('Male'))
Male2= len(Male1['SN'].unique())
MalePercent= round((Male2/total_count)*100,2)

#Percentage and Count of Female Players
Female1= data.groupby(['Gender']).get_group(('Female'))
Female2= len(Female1['SN'].unique())
FemalePercent= round((Female2/total_count)*100,2)

#Percentage and Count of Other / Non-Disclosed Players
Other1= data.groupby(['Gender']).get_group(('Other / Non-Disclosed'))
Other2= len(Other1['SN'].unique())
OtherPercent= round((Other2/total_count)*100,2)

#Make it into a Gender DataFrame
gender1 = {'Percent of Players':[MalePercent,FemalePercent,OtherPercent],'Ge
gender2 =  pd.DataFrame(gender1)
gender2= gender2.set_index('Gender')
gender2= gender2[['Gender Count','Percent of Players']]
gender2
```

Out[72]:

Gender	Gender Count	Percent of Players
Male	484	84.03
Female	81	14.06
Other	11	1.91

Purchasing Analysis (Gender)

- 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

** Purchasing Analysis (Gender)

In [73]:

```
#Purchase Count
FemalePurchaseCount = len(Female1)
MalePurchaseCount = len(Male1)
OtherPurchaseCount = len(Other1)

#Average Purchase Price
FemaleAvgPrice= round((Female1["Price"].sum())/len(Female1["Price"]),2)
MaleAvgPrice =round((Male1["Price"].sum())/len(Male1["Price"]),2)
OtherAvgPrice= round((Other1["Price"].sum())/len(Other1["Price"]),2)

#Total Purchase Value
FemaleTotalPurchase = round(Female1['Price'].sum(),2)
MaleTotalPurchase = round(Male1['Price'].sum(),2)
OtherTotalPurchase = round(Other1['Price'].sum(),2)

# Normalised Totals
# male/female/Other
NormFemale = round((FemaleTotalPurchase/FemalePurchaseCount), 2)
NormMale = round((MaleTotalPurchase/MalePurchaseCount), 2)
NormOther = round((OtherTotalPurchase/OtherPurchaseCount), 2)

ResBySex = {"Purchase Count":[FemalePurchaseCount, MalePurchaseCount, OtherP
                    "Gender":["Female","Male","Other / Non-Disclosed"],
                    "Average Purchase Price":[FemaleAvgPrice,MaleAvgPrice,Ot
                    "Total Purchase Value":[FemaleTotalPurchase,MaleTotalPur
                "Normalized Totals":[NormFemale,NormMale,NormOther]}
ResBySex1 = pd.DataFrame(ResBySex)
ResBySex1 = ResBySex1.set_index('Gender')
ResBySex1= ResBySex1[['Purchase Count','Average Purchase Price','Total Purch
ResBySex1
```

Out[73]:

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 [74]:
```
#Age Demographics
MaxAge = data['Age'].max()
print(MaxAge)
```

```
45
```

In [75]:
```
#Age Demographics
MinAge = data['Age'].min()
print(MinAge)
```

```
7
```

- Categorize the existing players using the age bins. Hint: use pd.cut()

pandas.cut(x, bins, right=True, labels=None, retbins=False, precision=3, include_lowest=False, duplicates='raise')

In [76]:
```
#bins of 4 years (i.e. <10, 10-14, 15-19, etc.)
bins = [0,10,14,19,24,29,34,39,46]
Agelabels = ["<10","10-14","15-19","20-24","25-29","30-34","35-39","40+"]
data['Age Summary'] = pd.cut(data['Age'],bins,labels= Agelabels)
print(bins)
```

```
[0, 10, 14, 19, 24, 29, 34, 39, 46]
```

In [77]:
```
data.columns
```

Out[77]:
```
Index(['Purchase ID', 'SN', 'Age', 'Gender', 'Item ID', 'Item Name', 'Price',
       'Age Summary'],
      dtype='object')
```

In [78]:
```
Total = data['Price'].sum()
print (Total)
```

```
2379.77
```

In [79]:
```
1 data.Price.sum
```

Out[79]:
```
<bound method Series.sum of 0      3.53
1      1.56
2      4.88
3      3.27
4      1.44
5      3.61
6      2.18
7      2.67
8      1.10
9      3.58
10     4.74
11     2.67
12     4.18
13     1.70
14     4.86
15     2.89
16     2.52
17     1.76
18     4.90
19     4.64
20     4.60
21     1.48
22     3.81
23     3.40
24     3.19
25     4.23
26     1.61
27     3.09
28     4.32
29     3.16
       ...
750    3.61
751    3.36
752    2.60
753    4.23
754    4.05
755    3.15
756    2.05
757    4.60
758    4.03
759    3.77
760    2.38
761    1.70
762    1.44
763    1.75
764    4.07
765    2.07
766    4.14
767    4.88
768    4.88
769    4.58
770    1.02
771    4.35
772    3.58
```

```
773    1.02
774    4.19
775    3.54
776    1.63
777    3.46
778    4.19
779    4.60
Name: Price, Length: 780, dtype: float64>
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [80]:
```
#Purchase Count
Bin1 = data.groupby(['Age Summary']).get_group(('<10'))
pc1 = len(Bin1['SN'].unique())
PerBin1 = (pc1/total_count)*100

Bin2 = data.groupby(['Age Summary']).get_group(('10-14'))
pc2 = len(Bin2['SN'].unique())
PerBin2 = (pc2/total_count)*100

Bin3 = data.groupby(['Age Summary']).get_group(('15-19'))
pc3 = len(Bin3['SN'].unique())
PerBin3 = (pc3/total_count)*100

Bin4 = data.groupby(['Age Summary']).get_group(('20-24'))
pc4 = len(Bin4['SN'].unique())
PerBin4 = (pc4/total_count)*100

Bin5 = data.groupby(['Age Summary']).get_group(('25-29'))
pc5 = len(Bin5['SN'].unique())
PerBin5 = (pc5/total_count)*100

Bin6 = data.groupby(['Age Summary']).get_group(('30-34'))
pc6 = len(Bin6['SN'].unique())
PerBin6 = (pc6/total_count)*100

Bin7 = data.groupby(['Age Summary']).get_group(('35-39'))
pc7 = len(Bin7['SN'].unique())
PerBin7 = (pc7/total_count)*100

Bin8 = data.groupby(['Age Summary']).get_group(('40+'))
pc8 = len(Bin8['SN'].unique())
PerBin8 = (pc8/total_count)*100
```

- Calculate the numbers and percentages by age group

In [81]:
```
PlayerBinsCount=[pc1,pc2,pc3,pc4,pc5,pc6,pc7,pc8]
PercentBins= [PerBin1,PerBin2,PerBin3,PerBin4,PerBin5,PerBin6,PerBin7,PerBin
PercentBins= [round(x,2) for x in PercentBins]
print(PlayerBinsCount)
```

```
[24, 15, 107, 258, 77, 52, 31, 12]
```

In [82]:
```
AgeDem = {"Age Summary":Agelabels,"Total Player Count":PlayerBinsCount,"Perc
AgeDem1 = pd.DataFrame(AgeDem)
AgeDem1 = AgeDem1.set_index('Age Summary')
AgeDem1
```

Out[82]:

Age Summary	Total Player Count	Percentage Of Players
<10	24	4.17
10-14	15	2.60
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 [83]:

```
#Age Demographics
MaxAge = data['Age'].max()


#Age Demographics
MinAge = data['Age'].min()


#bins of 4 years (i.e. <10, 10-14, 15-19, etc.)
bins = [0,10,14,19,24,29,34,39,46]
Agelabels = ["<10","10-14","15-19","20-24","25-29","30-34","35-39","40+"]
data['Age Summary'] = pd.cut(data['Age'],bins,labels= Agelabels)


PlayerBinsCount=[pc1,pc2,pc3,pc4,pc5,pc6,pc7,pc8]
PercentBins= [PerBin1,PerBin2,PerBin3,PerBin4,PerBin5,PerBin6,PerBin7,PerBin
PercentBins= [round(x,2) for x in PercentBins]


#Purchase Count
Bin1 = data.groupby(['Age Summary']).get_group(('<10'))
pc1 = len(Bin1['SN'].unique())
PerBin1 = (pc1/total_count)*100

Bin2 = data.groupby(['Age Summary']).get_group(('10-14'))
pc2 = len(Bin2['SN'].unique())
PerBin2 = (pc2/total_count)*100

Bin3 = data.groupby(['Age Summary']).get_group(('15-19'))
pc3 = len(Bin3['SN'].unique())
PerBin3 = (pc3/total_count)*100

Bin4 = data.groupby(['Age Summary']).get_group(('20-24'))
pc4 = len(Bin4['SN'].unique())
PerBin4 = (pc4/total_count)*100

Bin5 = data.groupby(['Age Summary']).get_group(('25-29'))
pc5 = len(Bin5['SN'].unique())
PerBin5 = (pc5/total_count)*100

Bin6 = data.groupby(['Age Summary']).get_group(('30-34'))
pc6 = len(Bin6['SN'].unique())
PerBin6 = (pc6/total_count)*100

Bin7 = data.groupby(['Age Summary']).get_group(('35-39'))
pc7 = len(Bin7['SN'].unique())
PerBin7 = (pc7/total_count)*100

Bin8 = data.groupby(['Age Summary']).get_group(('40+'))
pc8 = len(Bin8['SN'].unique())
PerBin8 = (pc8/total_count)*100
```

In [84]:
```
1 data.columns
```

Out[84]:
```
Index(['Purchase ID', 'SN', 'Age', 'Gender', 'Item ID', 'Item Name', 'Price',
       'Age Summary'],
      dtype='object')
```

In [85]:
```
1 AgeDem = {"Age Summary":Agelabels,"Total Player Count":PlayerBinsCount,"Perc
2 AgeDem1 = pd.DataFrame(AgeDem)
3 AgeDem1 = AgeDem1.set_index('Age Summary')
4 AgeDem1
```

Out[85]:

Age Summary	Total Player Count	Percentage Of Players
<10	24	4.17
10-14	15	2.60
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

In [ ]:
```
1
```

In [ ]:
```
1
```

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 [86]:

```
#SN
SN = data.groupby(data["SN"])
ScreenName = SN["SN"].unique()

#Purchase Count
SNCount = SN['Age'].count()

#Average Purchase Price
SNAverage = round(SN['Price'].mean(),2)

#Total Purchase Value
SNTotal = SN['Price'].sum()


TopSpend = {"SN":ScreenName,"Purchase Count":SNCount,
                "Average Purchase Price":SNAverage,"Total Purchase Value":S
TopSpend1= pd.DataFrame(TopSpend)
TopSpend1= TopSpend1.set_index('SN')
TopSpend1 = TopSpend1.sort_values("Total Purchase Value",ascending=False)
TopSpend1 = TopSpend1[['Purchase Count', 'Average Purchase Price', 'Total Pu

TopSpend1.iloc[:5]
```

Out[86]:

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 [87]:

```
#Item ID
ItemId = data.groupby(data['Item ID'])
Items = ItemId['Item ID'].unique()
#Item Name

ItemName = ItemId["Item Name"].unique()

#Purchase Count
ItemPurCount = ItemId['Age'].count()

#Item Price
ItemPrice= ItemId['Price'].unique()


#Total Purchase Value
ItemTotalPurchase = ItemId['Price'].sum()

ItemTable = {'Item ID':Items,'Item Name':ItemName,'Item Price':ItemPrice,'It
ItemTable1 = pd.DataFrame(ItemTable)
ItemTable1 = ItemTable1.set_index('Item ID')
ItemTable1= ItemTable1.sort_values('Item Count', ascending=False)
ItemTable1 = ItemTable1[['Item Name','Item Count','Item Price','Total Purcha
ItemTable1.iloc[:5]
```

Out[87]:

Item ID	Item Name	Item Count	Item Price	Total Purchase
[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	[4.9]	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 [88]:
```
MostProfit= ItemTable1.sort_values('Total Purchase', ascending=False)
MostProfit[:5]
```

Out[88]:

Item ID	Item Name	Item Count	Item Price	Total Purchase
[178]	[Oathbreaker, Last Hope of the Breaking Storm]	12	[4.23]	50.76
[82]	[Nirvana]	9	[4.9]	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 [ ]:
```

```

In [ ]:
```

```

In [ ]:
```

```