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%).
- The age group that spends the most money is the 20-24 (46.8%) with $ 1,114.06 dollars as total purchase value and an average purchase of $ 4.32. In contrast, the demographic group that has the highest average purchase is the 35-39 (5.3%) with $ 4.76 and a total purchase value of $ 147.67.
- The average purchase per person is about $ 4.00 with the top spenders paying almost $ 19.00 for their purchase. However, we have 97% paying under $ 10.00.
- The company offers 183 items. The most popular and profitable one is "Oathbreaker, Last Hope of the Breaking Storm" with 12 buys (1.53%) and "Nirvana" and "Fiery Glass Crusader" with both of them with 9 buys (1.15%).

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 [42]:
```
# Dependencies and Setup
import pandas as pd
import numpy as np

# File to Load
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[42]:

	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]:
```
# create a dataframe
purchase_df = pd.DataFrame(purchase_data, columns=['Purchase ID', 'SN', 'Age', 
'Gender', 'Item ID', 'Item Name', 'Price'])

# player count
total_players_count = (purchase_df['SN'].nunique())
total_players_df = pd.DataFrame([total_players_count], columns=['Total Player
s'])
total_players_df
```

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]:
```
unique_items_total = purchase_df['Item ID'].nunique()

average_price = purchase_df['Price'].mean()
average_price_format = f'${average_price:.2f}'

number_of_purchases = purchase_df['Purchase ID'].count()

total_revenue = purchase_df['Price'].sum()
total_revenue_format = f'${total_revenue:,.2f}'

summary_df = pd.DataFrame([(unique_items_total, average_price_format, number_o
f_purchases, total_revenue_format)], columns=['Number of Unique Items', 'Avera
ge Price', 'Number of Purchases', 'Total Revenue'])
summary_df
```

Out[44]:

	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 [45]:
```
gender_groups = purchase_df.groupby(['Gender'])
gender_df = gender_groups['SN'].nunique()

summary_df = pd.DataFrame({'Total Count':gender_df})

percent = ((gender_df / gender_df.sum()) * 100)

summary_df['Percentage of Players'] = percent


summary_df = summary_df.sort_values(by=['Total Count'], ascending = False)
summary_df = summary_df.style.format({'Percentage of Players': '{:.2f}'})
summary_df.index.name = None
summary_df
```

Out[45]:

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

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 [46]:
```
total_count_gender = gender_groups.nunique()["SN"]

purchase_count = gender_groups["Purchase ID"].count()

avg_purchase_price = gender_groups["Price"].mean()

avg_purchase_total = gender_groups["Price"].sum()

avg_purchase_per_person = avg_purchase_total/total_count_gender

gender_demographics = pd.DataFrame({"Purchase Count": purchase_count, "Average
Purchase Price": avg_purchase_price, "Total Purchase Value":avg_purchase_total
, "Avg Total Purchase per Person": avg_purchase_per_person})

gender_demographics.index.name = "Gender"

gender_demographics.style.format({"Average Purchase Price":"${:,.2f}", "Total
 Purchase Value":"${:,.2f}", "Avg Total Purchase per Person":"${:,.2f}"})
```

Out[46]:

Gender	Purchase Count	Average Purchase Price	Total Purchase Value	Avg Total Purchase 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 [47]:
```
age_bins = [0, 9.99, 14.99, 19.99, 24.99, 29.99, 34.99, 39.99, 99999]
group_names = ["<10", "10-14", "15-19", "20-24", "25-29", "30-34", "35-39", "4
0+"]

purchase_data["Age Group"] = pd.cut(purchase_data["Age"],age_bins, labels=grou
p_names)
age_grouped = purchase_data.groupby("Age Group")
total_count_age = age_grouped["SN"].nunique()
percentage_by_age = (total_count_age/total_players_count) * 100

age_demographics = pd.DataFrame({"Total Count": total_count_age, "Percentage o
f Players": percentage_by_age})


age_demographics.style.format({"Percentage of Players":"{:,.2f}"})
```

Out[47]:

	Total Count	Percentage of Players
Age Group		
<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 [48]:
```
age_demographics_alt = pd.DataFrame(age_demographics)

purchase_count_age = age_grouped["Purchase ID"].count()

avg_purchase_price_age = age_grouped["Price"].mean()
total_purchase_value = age_grouped["Price"].sum()
avg_purchase_per_person_age = total_purchase_value/total_count_age
age_demographics_alt = pd.DataFrame({"Purchase Count": purchase_count_age, "Av
erage Purchase Price": avg_purchase_price_age, "Total Purchase Value":total_pu
rchase_value, "Avg Total Purchase per Person": avg_purchase_per_person_age})

age_demographics_alt.index.name = None

age_demographics_alt.style.format({"Average Purchase Price":"${:,.2f}", "Total
Purchase Value":"${:,.2f}", "Avg Total Purchase per Person":"${:,.2f}"})
new_output = age_demographics_alt.index.tolist()
under_10 = new_output.pop(0)
realligned_table = age_demographics_alt.reindex(new_output+[under_10])
realligned_table.style.format({"Average Purchase Price":"${:,.2f}", "Total Pur
chase Value":"${:,.2f}", "Avg Total Purchase per Person":"${:,.2f}"})
```

Out[48]:

	Purchase Count	Average Purchase Price	Total Purchase Value	Avg Total Purchase per Person
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
<10	23	$3.35	$77.13	$4.54

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 [49]:
```
# Group purchase data by screen names
spender_stats = purchase_data.groupby("SN")

# Count the total purchases by name
purchase_count_spender = spender_stats["Purchase ID"].count()

# Calculate the average purchase by name
avg_purchase_price_spender = spender_stats["Price"].mean()

# Calculate purchase total
total_purchase_spender = spender_stats["Price"].sum()

# Create data frame with obtained values
top_spenders = pd.DataFrame({"Purchase Count": purchase_count_spender, "Averag
e Purchase Price": avg_purchase_price_spender, "Total Purchase Value":total_pu
rchase_spender})

# Sort in descending order to obtain top 5 spender names
formatted_spenders = top_spenders.sort_values(["Total Purchase Value"], ascend
ing=False).head()

# Format with currency style
formatted_spenders.style.format({"Average Purchase Price":"${:,.2f}", "Total P
urchase Value":"${:,.2f}"})
```

Out[49]:

	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 [50]:
```
# Create new data frame with items related information
items = purchase_data[["Item ID", "Item Name", "Price"]]

# Group the item data by item id and item name
item_stats = items.groupby(["Item ID","Item Name"])

# Count the number of times an item has been purchased
purchase_count_item = item_stats["Price"].count()

# Calcualte the purchase value per item
purchase_value = (item_stats["Price"].sum())

# Find individual item price
item_price = purchase_value/purchase_count_item

# Create data frame with obtained values
most_popular_items = pd.DataFrame({"Purchase Count": purchase_count_item, "Ite
m Price": item_price, "Total Purchase Value":purchase_value})

# Sort in descending order to obtain top spender names and provide top 5 item
 names
popular_formatted = most_popular_items.sort_values(["Purchase Count"], ascendi
ng=False).head()

# Format with currency style
popular_formatted.style.format({"Item Price":"${:,.2f}", "Total Purchase Valu
e":"${:,.2f}"})
```

Out[50]:

		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 [41]:
```
# Sort the above table by total purchase value in descending order
popular_formatted = most_popular_items.sort_values(["Total Purchase Value"], a
scending=False).head()

# Format with currency style
popular_formatted.style.format({"Item Price":"${:,.2f}", "Total Purchase Valu
e":"${:,.2f}"})
```

Out[41]:

		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

In [ ]: