Heroes of Pymoli Data Analysis

- 84% of gamers are male
- 45% of gamers have 20-24 years and represents 47% of total revenue
- Although male are mostly the buyers, the average total purchase per person indicates female and male are very similar
- Despite of Extraction, Quickblade Of Trembling Hands is one of the most popular games (3 out of 5), It isn't in the list of the most profitable games

In [1]:
```
#Call dependences
import pandas as pd
import os
#Setup file location
csv_file_directory = "Resources"
csv_file_name = "purchase_data.csv"
csv_file_path = os.path.join(csv_file_directory, csv_file_name)
#Load dataframe
purchase_df = pd.read_csv(csv_file_path)

#Section: Player count
def get_total_players(df):
    total_players = pd.DataFrame([{
    'Total Players':len(df.groupby('SN'))
}])
    return total_players

#Section: Purchasing Analysis (Total)
def get_summary_purchase():
    summary_unique_items = len(purchase_df.groupby('Item ID'))
    summary_average_price = purchase_df["Price"].mean()
    summary_number_purchases = purchase_df["Purchase ID"].count()
    summary_total_revenue = purchase_df['Price'].sum()
    summary_purchase = pd.DataFrame([{
    'Number of Unique Items' : summary_unique_items,
    'Average Price' : '${:,.2f}'.format(summary_average_price),
    'Number of Purchases' : summary_number_purchases,
    'Total revenue' : '${:,.2f}'.format(summary_total_revenue)
    }])
    summary_purchase = summary_purchase[['Number of Unique Items',
                                         'Average Price',
                                         'Number of Purchases',
                                         'Total revenue']]

    return summary_purchase

#Section: Gender Demographics
def get_gender_demographics(df):
    count_female_gender = len(df[(df['Gender']== 'Female')])
    count_male_gender = len(df[(df['Gender']=='Male')])
    count_other_gender = len(df[(df['Gender'] != 'Female') & (df['Gender']
    count_total_gender = count_female_gender+count_male_gender+count_other_
    #Percentage per gender
    percentage_female_gender = count_female_gender / count_total_gender * 1
    percentage_male_gender = count_male_gender / count_total_gender * 100
    percentage_other_gender = count_other_gender / count_total_gender * 100

    gender_demographics = pd.DataFrame({
        'Total Count': {
            'Male': count_male_gender,
            'Female': count_female_gender,
            'Other / Non-Disclosed': count_other_gender
        },
        'Percentage of Players':{
            'Male' : '{:,.2f}'.format(percentage_male_gender),
            'Female' : '{:,.2f}'.format(percentage_female_gender),
            'Other / Non-Disclosed' : '{:,.2f}'.format(percentage_other_gen
```

```
        }
    })

    return gender_demographics

#Section: Purchasing Analysis (Gender)
def purchase_detail_by_gender(df, gender):
    #Filter dataframe by gender
    gender_purchases = df[(df['Gender']==gender)]
    #Get purchases history by gender
    gender_purchase_count = len(gender_purchases)
    gender_avg_purchase_price = gender_purchases['Price'].mean()
    gender_purchase_value = gender_purchases['Price'].sum()
    #
    gender_purchases = gender_purchases.groupby('SN').sum()
    gender_purchases = gender_purchases['Price'].mean()
    #return values as a list
    return [gender_purchase_count, gender_avg_purchase_price, gender_purcha

def purchasing_analysis_gender(df):
    #Gender Purchase History
    female_list = purchase_detail_by_gender(df,'Female')
    male_list = purchase_detail_by_gender(df,'Male')
    other_list = purchase_detail_by_gender(df,'Other / Non-Disclosed')

    purchase_analysis_df = pd.DataFrame({
        'Purchase Count' : {
            'Female' : female_list[0],
            'Male' : male_list[0],
            'Other / Non-Disclosed' : other_list[0]
        },
        'Average Purchases Price' : {
            'Female' : '${:,.2f}'.format(female_list[1]),
            'Male' : '${:,.2f}'.format(male_list[1]),
            'Other / Non-Disclosed' : '${:,.2f}'.format(other_list[1])
        },
        'Total Purchase Value' : {
            'Female' : '${:,.2f}'.format(female_list[2]),
            'Male' : '${:,.2f}'.format(male_list[2]),
            'Other / Non-Disclosed' : '${:,.2f}'.format(other_list[2])
        },
        'Avg Total Purchase per Person' : {
            'Female' : '${:,.2f}'.format(female_list[3]),
            'Male' : '${:,.2f}'.format(male_list[3]),
            'Other / Non-Disclosed' : '${:,.2f}'.format(other_list[3])
        }
    })
    return purchase_analysis_df

#Section: Age Demographics
def age_demo_bins(df):
    #Set up bins
    bins = [0,9,14,19,24,29,34,39,45]
    bins_label = ['<10','10-14','15-19','20-24','25-29','30-34','35-39', '4
    return pd.cut(df['Age'], bins, labels=bins_label)

def age_demographics(df):
```

```
    #Set dataframe up
    age_demo_df = df

    age_demo_df[''] = age_demo_bins(age_demo_df)
    age_demo_df = age_demo_df.groupby('').count()
    age_demo_df = age_demo_df[['SN']]
    #Get Percentage of Players
    percentage_player = lambda x: x / 576 * 100 if x['SN'] > 0 else ''
    age_demo_df['Percentage of Players'] = age_demo_df.apply(percentage_pla
    #age_demo_df['Percentage of Players'] = age_demo_df['Percentage of Play
    age_demo_df['Percentage of Players'] = put_format_rows(age_demo_df, 'Pe
    age_demo_df.columns = ['Total Count','Percentage of Players']

    return age_demo_df

#Section: Purchasing Analysis (Age)
def purchasing_analysis(df):
    paa_df = df
    paa_df['age_cat'] = age_demo_bins(paa_df)
    #paa_atpp used to build Avg Total Purchase per person
    paa_atpp = paa_df
    paa_atpp = paa_atpp.groupby(['age_cat','SN'])['Price'].sum().reset_inde
    paa_atpp = paa_atpp.groupby('age_cat')['Price'].mean().reset_index()
    #paa_df2 used to build the rest of dataframe
    paa_df2 = df.groupby(['age_cat']).agg({'Price' : ['count','mean','sum']
    paa_df2.columns = paa_df2.columns.droplevel(0)
    paa_df2.columns = ['age_cat', 'Purchase Count', 'Average Purchase Price
    #Merge dataframe to display summary
    paa_df2 = paa_df2.merge(paa_atpp, on='age_cat')
    #Adjust columns names to set up index
    paa_df2.columns = ['', 'Purchase Count', 'Average Purchase Price', 'Tot
    #Formatting rows
    paa_df2['Average Purchase Price'] = put_format_rows(paa_df2, 'Average P
    paa_df2['Total Purchase Value'] = put_format_rows(paa_df2, 'Total Purch
    paa_df2['Avg Total Purchase per Person'] = put_format_rows(paa_df2, 'Av
    #Return dataframe setting up index
    return paa_df2.set_index('')

#Section: Top Spender
def top_spenders(df):
    top_spender_df = df.groupby('SN').agg({'Purchase ID': 'size', 'Price':[
    top_spender_df.columns = top_spender_df.columns.droplevel(0)
    top_spender_df.columns = ['SN', 'Purchase Count', 'Average Purchase Pri
    #Sorting values
    top_spender_df = top_spender_df.sort_values(by='Total Purchase Value',
    #Formatting rows
    top_spender_df['Average Purchase Price'] = put_format_rows(top_spender_
    top_spender_df['Total Purchase Value'] = put_format_rows(top_spender_df
    #Return dataframe ready to be displayed
    return top_spender_df

#Section: Most Popular Items and Most Profitable Itmes. Var order is used t
def most_popular_items(df, order):
    most_pop_item = df
    most_pop_item = most_pop_item.groupby(['Item ID','Item Name']).agg({'Pr
    most_pop_item.columns = most_pop_item.columns.droplevel(0)
    most_pop_item['Total Purchase Value'] = most_pop_item['count'] * most_p
```

```
    most_pop_item.columns = ['Item ID', 'Item Name', 'Purchase Count', 'Ite
    most_pop_item = most_pop_item.set_index(['Item ID', 'Item Name'])
    #Sorting dataframe according of "order" argument
    most_pop_item = most_pop_item.sort_values(order, ascending=False).head(
    #Formatting rows
    most_pop_item['Item Price'] = put_format_rows(most_pop_item, 'Item Pric
    most_pop_item['Total Purchase Value'] = put_format_rows(most_pop_item,
    #Return dataframe ready to be displayed
    return most_pop_item

#Special function to format any row of any dataframe. Two types_format, mon
def put_format_rows(df, row, type_format):
    if type_format == 'money':
        type_format = '${:,.2f}'
    elif type_format == 'float':
        type_format = '{:,.2f}'

    df[row] = df[row].map(type_format.format)
    return df[row]
```

In [2]:
```
purchase_df.head()
```

Out[2]:

	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 of players

In [3]:
```
#Counting total players
get_total_players(purchase_df)
```

Out[3]:

	Total Players
0	576

Purchasing Analysis (Total)

- Run basics 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 summay data frame

In [4]:
```
#Summary Purchasing Analysis
get_summary_purchase()
```

Out[4]:

	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
- Peecentage and Count of Other / Non-Disclosed

In [5]:
```
#Gender demographics
get_gender_demographics(purchase_df.drop_duplicates(['SN']))
```

Out[5]:

	Total Count	Percentage of Players
Female	81	14.06
Male	484	84.03
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 [6]:
```
#Call purchasing_analysis_gender function, df filtered as parameter
purchasing_analysis_gender(purchase_df)
```

Out[6]:

	Purchase Count	Average Purchases 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 [7]:
```
#Call age_demographics function, grouping df by SN, Gender and Age
age_demographics(purchase_df.groupby(['SN','Gender','Age']).count().reset_i
```

Out[7]:

	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 [8]:
```
#Call purchasing_analysis to create bin and run basic calculations to get f
purchasing_analysis(purchase_df)
```

Out[8]:

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

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 [9]:
```
#Call top_spender to get data and display summary
top_spenders(purchase_df)
```

Out[9]:

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 [10]:
```
most_popular_items(purchase_df, order = 'Purchase Count')
```

Out[10]:

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 [11]:
```
most_popular_items(purchase_df, order = 'Total Purchase Value')
```

Out[11]:

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 [ ]:
```
```