In [ ]:

```
#Three observable trends from the data
#1-majority of players are male (approximately 84%)
#2-almost half of all players fall between the ages of 20-24 (approximately 45%)
#3-
```

In [1]:

```
#import dependencies
import pandas as pd
import numpy as py
```

In [3]:

```
file = 'purchase_data.csv'
purchase_data = pd.read_csv(file)
```

In [5]:

```
#Player Count

player_demographics = purchase_data.loc[:, ["Gender", "SN", "Age"]]
player_demographics = player_demographics.drop_duplicates()
num_players = player_demographics.count()[0]

# Display the total number of players
pd.DataFrame({"Total Players": [num_players]})
```

Out[5]:

| | Total Players |
|---|---|
| **0** | 576 |

In [6]:

```
#Purchasing Analysis (Total)

# Run basic calculations
average_item_price = purchase_data["Price"].mean()
total_purchase_value = purchase_data["Price"].sum()
purchase_count = purchase_data["Price"].count()
item_count = len(purchase_data["Item ID"].unique())

# Create a DataFrame to hold results
summary_table = pd.DataFrame({"Number of Unique Items": item_count,
                              "Total Revenue": [total_purchase_value],
                              "Number of Purchases": [purchase_count],
                              "Average Price": [average_item_price]})

# Minor Data Munging
summary_table = summary_table.round(2)
summary_table ["Average Price"] = summary_table["Average Price"].map("${:,.2f}".forr
summary_table ["Number of Purchases"] = summary_table["Number of Purchases"].map("{
summary_table ["Total Revenue"] = summary_table["Total Revenue"].map("${:,.2f}".forr
summary_table = summary_table.loc[:,["Number of Unique Items", "Average Price", "Nur

# Display the summary_table
summary_table
```

Out[6]:

| | Number of Unique Items | Average Price | Number of Purchases | Total Revenue |
|---|---|---|---|---|
| **0** | 183 | $3.05 | 780 | $2,379.77 |

In [7]:

```
#Gender Demographics

# Calculate the Number and Percentage by Gender
gender_demographics_totals = player_demographics["Gender"].value_counts()
gender_demographics_percents = gender_demographics_totals / num_players * 100
gender_demographics = pd.DataFrame({"Total Count": gender_demographics_totals, "Perc

# Minor Data Munging
gender_demographics = gender_demographics.round(2)

gender_demographics
```

Out[7]:

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

In [10]:

```
gender_purchase_total = purchase_data.groupby(["Gender"]).sum()["Price"].rename("Tot
gender_average = purchase_data.groupby(["Gender"]).mean()["Price"].rename("Average I
gender_counts = purchase_data.groupby(["Gender"]).count()["Price"].rename("Purchase

# Calculate Normalized Purchasing
avg_total_per_person = gender_purchase_total / gender_demographics["Total Count"]

# Convert to DataFrame
gender_data = pd.DataFrame({"Purchase Count": gender_counts, "Average Purchase Price

# Minor Data Munging
gender_data["Average Purchase Price"] = gender_data["Average Purchase Price"].map("$
gender_data["Total Purchase Value"] = gender_data["Total Purchase Value"].map("${:,
gender_data ["Purchase Count"] = gender_data["Purchase Count"].map("{:,}".format)
gender_data["Avg Total Purchase per Person"] = gender_data["Avg Total Purchase per I
gender_data = gender_data.loc[:, ["Purchase Count", "Average Purchase Price", "Total

# Display the Gender Table
gender_data
```

Out[10]:

| 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 |

In [11]:

```
#Age Demographics
# Establish the bins
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+"]

# Categorize the existing players using the age bins
player_demographics["Age Ranges"] = pd.cut(player_demographics["Age"], age_bins, lab

# Calculate the Numbers and Percentages by Age Group
age_demographics_totals = player_demographics["Age Ranges"].value_counts()
age_demographics_percents = age_demographics_totals / num_players * 100
age_demographics = pd.DataFrame({"Total Count": age_demographics_totals, "Percentage

# Minor Data Munging
age_demographics = age_demographics.round(2)

# Display Age Demographics Table
age_demographics.sort_index()
```

Out[11]:

| | 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 |

In [12]:

```
# Bin the Purchasing Data
purchase_data["Age Ranges"] = pd.cut(purchase_data["Age"], age_bins, labels=group_na

# Run basic calculations
age_purchase_total = purchase_data.groupby(["Age Ranges"]).sum()["Price"].rename("To
age_average = purchase_data.groupby(["Age Ranges"]).mean()["Price"].rename("Average
age_counts = purchase_data.groupby(["Age Ranges"]).count()["Price"].rename("Purchase

# Calculate Normalized Purchasing
normalized_total = age_purchase_total / age_demographics["Total Count"]

# Convert to DataFrame
age_data = pd.DataFrame({"Purchase Count": age_counts, "Average Purchase Price": age

# Minor Data Munging
age_data["Average Purchase Price"] = age_data["Average Purchase Price"].map("${:,.2f
age_data["Total Purchase Value"] = age_data["Total Purchase Value"].map("${:,.2f}".f
age_data ["Purchase Count"] = age_data["Purchase Count"].map("{:,}".format)
age_data["Normalized Totals"] = age_data["Normalized Totals"].map("${:,.2f}".format)
age_data = age_data.loc[:, ["Purchase Count", "Average Purchase Price", "Total Purch

# Display the Age Table
age_data
```

Out[12]:

| | Purchase Count | Average Purchase Price | Total Purchase Value | Normalized Totals |
|---|---|---|---|---|
| **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 |

In [36]:

```
#Top Spenders

# Basic Calculations
user_total = purchase_data.groupby(["SN"]).sum()["Price"].rename("Total Purchase Val
user_average = purchase_data.groupby(["SN"]).mean()["Price"].rename("Average Purchas
user_count = purchase_data.groupby(["SN"]).count()["Price"].rename("Purchase Count"

# Convert to DataFrame
user_data = pd.DataFrame({"Total Purchase Value": user_total, "Average Purchase Pric

# Minor Data Munging
user_data["Average Purchase Price"] = user_data["Average Purchase Price"].map("${:,
user_data["Total Purchase Value"] = user_data["Total Purchase Value"].map("${:,.2f}"
user_data = user_data.loc[:,["Purchase Count", "Average Purchase Price", "Total Purc


# Display Table
user_data.sort_values("Total Purchase Value", ascending = False).head()
```

Out[36]:

| SN | Purchase Count | Average Purchase Price | Total Purchase Value |
|---|---|---|---|
| **Haillyrgue51** | 3 | $3.17 | $9.50 |
| **Phistym51** | 2 | $4.75 | $9.50 |
| **Lamil79** | 2 | $4.64 | $9.29 |
| **Aina42** | 3 | $3.07 | $9.22 |
| **Saesrideu94** | 2 | $4.59 | $9.18 |

In [38]:

```
#Most Popular Items

# Extract item Data
item_data = purchase_data.loc[:,["Item ID", "Item Name", "Price"]]

# Perform basic calculations
total_item_purchase = item_data.groupby(["Item ID", "Item Name"]).sum()["Price"].ren
average_item_purchase = item_data.groupby(["Item ID", "Item Name"]).mean()["Price"]
item_count = item_data.groupby(["Item ID", "Item Name"]).count()["Price"].rename("Pu

# Minor Data Munging
item_data_pd = pd.DataFrame({"Total Purchase Value": total_item_purchase, "Item Pric
item_data_pd["Item Price"] = item_data_pd["Item Price"].map("${:,.2f}".format)
item_data_pd ["Purchase Count"] = item_data_pd["Purchase Count"].map("{:,}".format)
item_data_pd["Total Purchase Value"] = item_data_pd["Total Purchase Value"].map("${
item_data_pd = item_data_pd.loc[:,["Purchase Count", "Item Price", "Total Purchase V

# Display the Item Table
item_data_pd.sort_values("Purchase Count", ascending=False).head(10)
```

Out[38]:

| Item ID | Item Name | Purchase Count | Item Price | Total Purchase Value |
|---|---|---|---|---|
| **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 |
| **92** | **Final Critic** | 8 | $4.88 | $39.04 |
| **75** | **Brutality Ivory Warmace** | 8 | $2.42 | $19.36 |
| **59** | **Lightning, Etcher of the King** | 8 | $4.23 | $33.84 |
| **19** | **Pursuit, Cudgel of Necromancy** | 8 | $1.02 | $8.16 |
| **60** | **Wolf** | 8 | $3.54 | $28.32 |
| **34** | **Retribution Axe** | 8 | $2.22 | $17.76 |
| **72** | **Winter's Bite** | 8 | $3.77 | $30.16 |

In [37]:

```
#most profitable items
# Display the Item Table (Sorted by Total Purchase Value)
item_data_pd.sort_values("Total Purchase Value", ascending=False).head(5)
```

Out[37]:

| Item ID | Item Name | Purchase Count | Item Price | Total Purchase Value |
|---|---|---|---|---|
| **63** | **Stormfury Mace** | 2 | $4.99 | $9.98 |
| **29** | **Chaos, Ender of the End** | 5 | $1.98 | $9.90 |
| **173** | **Stormfury Longsword** | 2 | $4.93 | $9.86 |
| **1** | **Crucifer** | 3 | $3.26 | $9.78 |
| **38** | **The Void, Vengeance of Dark Magic** | 4 | $2.37 | $9.48 |

In [ ]: