Three observable trends based on the data

- People between 20-24 years of age represent more than half of the players. Folowed by the immediate lower and above age brackets. Data seems to be normally distributed.
- More than 3/4 of the players are male, acounting for ~84% of the players. However, females spend ~6% more than males.
- The average price per game is $3.05, but the most popular items are, mostly, 50% more expensive than the average.

In [111]:

```
# Dependencies and Setup
import pandas as pd

# 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)
```

Player Count

- Display the total number of players

In [112]:

```
purchase_data.head()
```

Out[112]:

	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

In [113]:

```
PlayerCount = len(purchase_data["SN"].unique())
playercount_disp = pd.DataFrame({"Player Count": [PlayerCount]})
playercount_disp
```

Out[113]:

	Player Count
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 [125]:

```python
#Get the info
unique_items = purchase_data["Item ID"].nunique()
average_price = purchase_data["Price"].mean()
number_of_purchases = purchase_data["Purchase ID"].count()
total_revenue = purchase_data["Price"].sum()

#create dataframe (aka as make pretty)
purchasing_summary = pd.DataFrame({"Number of Unique Items":[unique_items],
                                   "Average Price":[average_price],
                                   "Number of Purchases":[number_of_purchases],
                                   "Total Revenue":[total_revenue]})

#arrange in dataframe (aka as make pretty)
purchasing_summary_table = pd.DataFrame(purchasing_summary, columns=["Number of Unique Items","Average Price", "Number of Purchases", "Total Revenue"])

#round the numbers to 2 decimal places
purchasing_summary_table = purchasing_summary_table.round(2)

#print the table
purchasing_summary_table.head()
```

Out[125]:

	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 [126]:

```
#get the info
players_count = purchase_data["Gender"].count()
gender_count = purchase_data["Gender"].value_counts()
demographic_percentage = gender_count/players_count*100

#arrange in dataframe (aka as make pretty)
player_demographics = pd.DataFrame({"Total Count": gender_count, "Percentage of Players": demographic_percentage})

player_demographics["Percentage of Players"] = player_demographics["Percentage of Players"].map("{:,.2f}%".format)

#print the table
player_demographics.head()
```

Out[126]:

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

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 [127]:

```
purchase_count = purchase_data.groupby(['Gender']).count()['Purchase ID'
]

average_price = purchase_data.groupby(['Gender']).mean()['Price']

purchase_value = purchase_data.groupby(['Gender']).sum()['Price']

average_total = purchase_value / player_demographics['Total Count']

gender_demographics = pd.DataFrame({"Purchase Count" : purchase_count.ma
p("{:,}".format),
                                    "Average Purchase Price" : average_p
rice.map("${:,.2f}".format),
                                    "Total Purchase Value" : purchase_va
lue.map("${:,.2f}".format),
                                    "Avg Total Purchase per Person": ave
rage_total.map("${:,.2f}".format)})

gender_demographics.head()
```

Out[127]:

	Purchase Count	Average Purchase Price	Total Purchase Value	Avg Total Purchase pPerson
Gender				
Female	113	$3.20	$361.94	$3.20
Male	652	$3.02	$1,967.64	$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 [117]:

```
# Figure out the minimum and maximum age
print(purchase_data["Age"].max())
print(purchase_data["Age"].min())
```

```
45
7
```

In [130]:

```
bins = [0, 9, 14, 19, 24, 29, 34, 39, 50]

# Create labels for the bins
group_labels = ["<10", "10-14", "15-19", "20-24", "25-29", "30-34", "35-
39", "40+"]

player_bins["Age Ranges"] = pd.cut(purchase_data["Age"], bins = bins, la
bels = group_labels)

age_groups = purchase_data.groupby("Age")

age_demographic_counts = player_bins["Age Ranges"].value_counts()
age_demographics_pct = age_demographic_counts / PlayerCount * 100
age_demographics_pct = age_demographics_pct.map("{:,.2f}%".format)

age_demographics = pd.DataFrame({"Total Count": age_demographic_counts,
                                 "Percentage of Players": age_demographi
cs_pct})

age_demographics = age_demographics.round(2)
age_demographics.sort_index()
```

Out[130]:

	Total Count	Percentage of Players
<10	23	3.99%
10-14	28	4.86%
15-19	136	23.61%
20-24	365	63.37%
25-29	101	17.53%
30-34	73	12.67%
35-39	41	7.12%
40+	13	2.26%

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 [131]:

```
bins = [0, 9, 14, 19, 24, 29, 34, 39, 50]

# Create labels for the bins
group_labels = ["<10", "10-14", "15-19", "20-24", "25-29", "30-34", "35-
39", "40+"]

player_bins["Age Ranges"] = pd.cut(purchase_data["Age"], bins = bins, la
bels = group_labels)

age_groups = purchase_data.groupby("Age")

# get per age: count, sum and mean of purchases and purchase value
count_per_age = age_groups["Age"].count()
average_per_age = age_groups["Price"].mean()
sum_per_age = age_groups["Price"].sum()

# get the average purchase per person/per age

avg_purchase_person_age = sum_per_age/PlayerCount

purchase_age_demographics = pd.DataFrame({"Purchase Count" : count_per_a
ge.map("{:,}".format),
                                    "Average Purchase Price" : average_p
er_age.map("${:,.2f}".format),
                                    "Total Purchase Value" : sum_per_age
.map("${:,.2f}".format),
                                    "Avg Total Purchase pPerson": avg_pu
rchase_person_age.map("${:,.2f}".format)})

purchase_age_demographics
purchase_age_demographics.sort_index()
```

Out[131]:

	Purchase Count	Average Purchase Price	Total Purchase Value	Avg Total Purchase pPerson
Age				
7	9	$3.65	$32.89	$0.06
8	8	$3.25	$25.97	$0.05
9	6	$3.04	$18.27	$0.03
10	9	$3.54	$31.83	$0.06
11	7	$2.68	$18.79	$0.03
12	6	$2.63	$15.80	$0.03
13	4	$2.36	$9.45	$0.02
14	2	$3.46	$6.91	$0.01
15	35	$3.02	$105.65	$0.18
16	30	$3.02	$90.56	$0.16
17	22	$2.93	$64.48	$0.11
18	26	$3.16	$82.22	$0.14
19	23	$3.04	$69.98	$0.12
20	99	$3.17	$314.32	$0.55
21	62	$2.92	$180.74	$0.31
22	70	$2.96	$206.85	$0.36
23	67	$3.01	$201.93	$0.35
24	67	$3.14	$210.22	$0.36
25	59	$3.08	$181.90	$0.32
26	14	$2.87	$40.19	$0.07
27	10	$2.72	$27.23	$0.05
28	5	$1.69	$8.45	$0.01
29	13	$2.71	$35.23	$0.06
30	35	$3.15	$110.32	$0.19
31	7	$3.44	$24.05	$0.04
32	8	$2.84	$22.69	$0.04
33	14	$2.49	$34.81	$0.06
34	9	$2.46	$22.13	$0.04
35	14	$3.72	$52.03	$0.09
36	5	$2.53	$12.66	$0.02
37	7	$3.64	$25.51	$0.04
38	9	$3.79	$34.15	$0.06
39	6	$3.89	$23.32	$0.04

	Purchase Count	Average Purchase Price	Total Purchase Value	Avg Total Purchase pPerson
Age				
40	6	$2.79	$16.71	$0.03
41	2	$3.27	$6.54	$0.01
42	1	$3.93	$3.93	$0.01
43	1	$4.00	$4.00	$0.01
44	2	$2.68	$5.36	$0.01
45	1	$1.70	$1.70	$0.00

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 [132]:

```
spenders = purchase_data.groupby(["SN"])

user_purchase_count = spenders["Purchase ID"].count()

user_avg_purchase = spenders["Price"].mean()

user_total_purchase = spenders["Price"].sum()


top_spenders = pd.DataFrame({"Purchase Count" : user_purchase_count,
                             "Average Purchase Price": user_avg_purchase
.map("${:,.2f}".format),
                             "Total Purchase Value" : user_total_purchas
e.map("${:,.2f}".format)})

sorted_spenders = top_spenders.sort_values(["Total Purchase Value"], asc
ending=False)

sorted_spenders.head()
```

Out[132]:

	Purchase Count	Average Purchase Price	Total Purchase Value
SN			
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

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 [133]:

```
item_list = purchase_data[["Item ID", "Item Name", "Price"]]

grouped_items = item_list.groupby(["Item ID", "Item Name"])

purchased_items = grouped_items["Price"].count()

item_value = grouped_items["Price"].sum()

item_price = item_value/purchased_items


most_popular_items = pd.DataFrame({"Purchase Count" : purchased_items,
                             "Item Price":  item_price.map("${:,.2f}".fo
rmat),
                             "Total Purchase Value" : item_value.map("$
{:,.2f}".format)})


most_popular_items = most_popular_items.sort_values(["Purchase Count"],
ascending=False)

most_popular_items.head()
```

Out[133]:

		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 [136]:

```
profitable_items = most_popular_items.sort_values(["Purchase Count", "Total Purchase Value"], ascending=False)

profitable_items.head()
```

Out[136]:

		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
108	**Extraction, Quickblade Of Trembling Hands**	9	$3.53	$31.77
19	**Pursuit, Cudgel of Necromancy**	8	$1.02	$8.16

In [ ]: