Observed Trends:

1. In both data sets, there are more men than women and other/non-disclosed genders purchasing items in the game.
2. In both data sets, the most items were purchased within the 20-24 age group
3. In both data sets, kids under 10 were purchasing more items than people over 40

In [1]:
```
#Dependencies
import pandas as pd
import numpy as np
import os
```

In [2]:
```
filename = input("file name:")
```

```
file name:purchase_data.json
```

In [3]:
```
# Set path for file
json_path = os.path.join("Resources", str(filename))
```

In [4]:
```
pymoli_data = pd.read_json(json_path)
pymoli_data.head()
```

Out[4]:

	Age	Gender	Item ID	Item Name	Price	SN
0	38	Male	165	Bone Crushing Silver Skewer	3.37	Aelalis34
1	21	Male	119	Stormbringer, Dark Blade of Ending Misery	2.32	Eolo46
2	34	Male	174	Primitive Blade	2.46	Assastnya25
3	21	Male	92	Final Critic	1.36	Pheusrical25
4	23	Male	63	Stormfury Mace	1.27	Aela59

In [5]:
```
# Player Count
player_count = pymoli_data["SN"].nunique()
player_frame = pd.DataFrame({"Total Players": [player_count]})
player_frame
```

Out[5]:

	Total Players
0	573

In [6]:

```
#Purchasing Analysis (Total)

#Number of Unique Items
unique_items= pymoli_data["Item ID"].nunique()

#Average Purchase Price
avg_price = round(pymoli_data["Price"].mean(),2)

#Total Number of Purchases
tot_purch = len(pymoli_data)

#Total Revenue
revenue = round(pymoli_data["Price"].sum(),2)

purch_analysis = pd.DataFrame(
    {"Number of Unique Items": [unique_items],
    "Average Price": [avg_price],
    "Number of Purchases": [tot_purch],
    "Total Revenue": [revenue]})
purch_analysis= purch_analysis[["Number of Unique Items","Average Price","Numb
er of Purchases","Total Revenue"]]
purch_analysis
```

Out[6]:

	Number of Unique Items	Average Price	Number of Purchases	Total Revenue
0	183	2.93	780	2286.33

In [7]:

```
#Gender Demographics

grouped_gender=pymoli_data.groupby(["Gender"])
count_gender = grouped_gender["SN"].nunique()

perc_gender = round((count_gender/player_count)*100,2)
gender = pd.DataFrame({"Total Count":count_gender,"Percentage of Players":perc
_gender})
gender
```

Out[7]:

	Percentage of Players	Total Count
Gender		
Female	17.45	100
Male	81.15	465
Other / Non-Disclosed	1.40	8

In [8]:

```
#Purchasing Analysis (Gender)
purchase_count = grouped_gender["SN"].count()
avg_price = round(grouped_gender["Price"].mean(),2)
tot_price = grouped_gender["Price"].sum()
norm_totals = round(tot_price/count_gender,2)
purch_analysis = pd.DataFrame({"Purchase Count": purchase_count,
                              "Average Purchase Price": avg_price,
                              "Total Purchase Value": tot_price,
                              "Normalized Totals": norm_totals})
purch_analysis = purch_analysis[["Purchase Count","Average Purchase Price", "T
otal Purchase Value", "Normalized Totals"]]
purch_analysis
```

Out[8]:

	Purchase Count	Average Purchase Price	Total Purchase Value	Normalized Totals
Gender				
Female	136	2.82	382.91	3.83
Male	633	2.95	1867.68	4.02
Other / Non-Disclosed	11	3.25	35.74	4.47

In [9]:

```
#Bins for Age Demographics
bins = [0]
bin_value = ["<10"]
max_age = pymoli_data["Age"].max()

for x in range(2,int(max_age/5)+2):
    bins = bins + [(x*5)-1]

for x in range(1,len(bins)-2):
    bin_value = bin_value + [str(bins[x]+1)+"-"+str(bins[x+1])]

bin_value = bin_value + [str(max_age) + "+"]
#print(bins)
#print(bin_value)

#Age Demographics - need to group by SN.. add back in age, gender.. groupby ag
e group (count and percentage)
pymoli_data["Age Group"] = pd.cut(pymoli_data["Age"],bins,labels = bin_value)
```

In [10]:

```
#Purchasing Analysis (Age)
age_group = pymoli_data.groupby(["Age Group"])
age_count = age_group.size()
age_avg_price = round(age_group["Price"].mean(),2)
age_tot_value = age_group["Price"].sum()
age_norm_tot = round(age_tot_value/age_count,2)
age_dataframe = pd.DataFrame({"Purchase Count":age_count,
                             "Average Purchase Price": age_avg_price,
                             "Total Purchase Value": age_tot_value,
                             "Normalized Totals": age_norm_tot})

age_dataframe = age_dataframe[["Purchase Count","Average Purchase Price",
"Total Purchase Value", "Normalized Totals"]]
age_dataframe
```

Out[10]:

	Purchase Count	Average Purchase Price	Total Purchase Value	Normalized Totals
Age Group				
<10	28	2.98	83.46	2.98
10-14	35	2.77	96.95	2.77
15-19	133	2.91	386.42	2.91
20-24	336	2.91	978.77	2.91
25-29	125	2.96	370.33	2.96
30-34	64	3.08	197.25	3.08
35-39	42	2.84	119.40	2.84
40-44	16	3.19	51.03	3.19
45+	1	2.72	2.72	2.72

In [11]:

```
top_spend_group = pymoli_data.groupby("SN")
top_spend_count = top_spend_group.size()
top_spend_avg = round(top_spend_group["Price"].mean(),2)
top_spenders = top_spend_group["Price"].sum()
top_spenders_frame = pd.DataFrame({"Total Purchase Value": top_spenders})

#Identify top 5 spenders
top_spenders_sort = top_spenders_frame.sort_values(by="Total Purchase Value",
                                                  ascending = False)
top_spenders_filter = top_spenders_sort.iloc[0:5].reset_index()

#join spenders with other calculations
top_spend_func_frame = pd.DataFrame({"Purchase Count": top_spend_count,
                                     "Average Purchase Price": top_spend_avg})
.reset_index()

top_spenders_final = pd.merge(top_spenders_filter, top_spend_func_frame, on="S
N")
top_spenders_final
```

Out[11]:

	SN	Total Purchase Value	Average Purchase Price	Purchase Count
0	Undirrala66	17.06	3.41	5
1	Saedue76	13.56	3.39	4
2	Mindimnya67	12.74	3.18	4
3	Haellysu29	12.73	4.24	3
4	Eoda93	11.58	3.86	3

In [12]:

```
top_items = pymoli_data.groupby("Item ID")
top_items_group2 = pymoli_data.groupby(["Item ID","Item Name","Price"])
top_items_count = top_items.size()
top_items_value = top_items_group2["Price"].sum()
top_items_frame = pd.DataFrame({"Purchase Count": top_items_count})
#Identify most popular items
top_items_sort = top_items_frame.sort_values(by="Purchase Count",
                                                  ascending = False).reset_
index()


end = top_items_sort["Purchase Count"].size

lastrow = 0

for row in range(4, end):
    if(top_items_sort.get_value(row,"Purchase Count") != top_items_sort.get
_value(row + 1,"Purchase Count")):
        lastrow = row
        break

if lastrow != 4:
    print("Note: As a result of ties, more than 5 items are the most popula
r. The tied values are printed")

top_items_filter = top_items_sort.iloc[0:lastrow+1].reset_index()

top_items_func_frame = pd.DataFrame({"Total Purchase Value": top_items_valu
e}).reset_index()

#merge tables
top_items_final = pd.merge(top_items_filter, top_items_func_frame, on="Item
 ID")

#fix column order and price name
top_items_final = top_items_final[["Item ID","Item Name","Purchase Count",
"Price","Total Purchase Value"]]
top_items_final = top_items_final.rename(columns={"Price":"Item Price"})
top_items_final
```

```
Note: As a result of ties, more than 5 items are the most popular. The tied v
alues are printed
```

Out[12]:

	Item ID	Item Name	Purchase Count	Item Price	Total Purchase Value
0	39	Betrayal, Whisper of Grieving Widows	11	2.35	25.85
1	84	Arcane Gem	11	2.23	24.53
2	31	Trickster	9	2.07	18.63
3	175	Woeful Adamantite Claymore	9	1.24	11.16
4	13	Serenity	9	1.49	13.41
5	34	Retribution Axe	9	4.14	37.26

In [13]:

```python
#Most Profitable Items
top_values_frame = pd.DataFrame({"Total Purchase Value": top_items_value})
top_values_sort = top_values_frame.sort_values(by="Total Purchase Value",
                                               ascending = False)
top_values_filter = top_values_sort.iloc[0:5].reset_index()
top_values_final = pd.merge(top_values_filter,top_items_sort, on="Item ID")

#rearrange columns and rename price column
top_values_final = top_values_final[["Item ID","Item Name","Purchase Count","P
rice","Total Purchase Value"]]
top_values_final = top_values_final.rename(columns={"Price":"Item Price"})
print()
top_values_final
```

Out[13]:

	Item ID	Item Name	Purchase Count	Item Price	Total Purchase Value
0	34	Retribution Axe	9	4.14	37.26
1	115	Spectral Diamond Doomblade	7	4.25	29.75
2	32	Orenmir	6	4.95	29.70
3	103	Singed Scalpel	6	4.87	29.22
4	107	Splitter, Foe Of Subtlety	8	3.61	28.88

In [14]:
```
#Dependencies
import pandas as pd
import numpy as np
import os
```

In [15]:
```
filename = input("file name:")
```

```
file name:purchase_data2.json
```

In [16]:
```
# Set path for file
json_path = os.path.join("Resources", str(filename))
```

In [17]:
```
pymoli_data = pd.read_json(json_path)
pymoli_data.head()
```

Out[17]:

	Age	Gender	Item ID	Item Name	Price	SN
0	20	Male	93	Apocalyptic Battlescythe	4.49	Iloni35
1	21	Male	12	Dawne	3.36	Aidaira26
2	17	Male	5	Putrid Fan	2.63	Irim47
3	17	Male	123	Twilight's Carver	2.55	Irith83
4	22	Male	154	Feral Katana	4.11	Philodil43

In [18]:
```
# Player Count
player_count = pymoli_data["SN"].nunique()
player_frame = pd.DataFrame({"Total Players": [player_count]})
player_frame
```

Out[18]:

	Total Players
0	74

In [19]:

```
#Purchasing Analysis (Total)

#Number of Unique Items
unique_items= pymoli_data["Item ID"].nunique()

#Average Purchase Price
avg_price = round(pymoli_data["Price"].mean(),2)

#Total Number of Purchases
tot_purch = len(pymoli_data)

#Total Revenue
revenue = round(pymoli_data["Price"].sum(),2)

purch_analysis = pd.DataFrame(
    {"Number of Unique Items": [unique_items],
    "Average Price": [avg_price],
    "Number of Purchases": [tot_purch],
    "Total Revenue": [revenue]})
purch_analysis= purch_analysis[["Number of Unique Items","Average Price","Numb
er of Purchases","Total Revenue"]]
purch_analysis
```

Out[19]:

	Number of Unique Items	Average Price	Number of Purchases	Total Revenue
0	64	2.92	78	228.1

In [20]:

```
#Gender Demographics

grouped_gender=pymoli_data.groupby(["Gender"])
count_gender = grouped_gender["SN"].nunique()

perc_gender = round((count_gender/player_count)*100,2)
gender = pd.DataFrame({"Total Count":count_gender,"Percentage of Players":perc
_gender})
gender
```

Out[20]:

	Percentage of Players	Total Count
Gender		
Female	17.57	13
Male	81.08	60
Other / Non-Disclosed	1.35	1

In [21]:

```
#Purchasing Analysis (Gender)
purchase_count = grouped_gender["SN"].count()
avg_price = round(grouped_gender["Price"].mean(),2)
tot_price = grouped_gender["Price"].sum()
norm_totals = round(tot_price/count_gender,2)
purch_analysis = pd.DataFrame({"Purchase Count": purchase_count,
                              "Average Purchase Price": avg_price,
                              "Total Purchase Value": tot_price,
                              "Normalized Totals": norm_totals})
purch_analysis = purch_analysis[["Purchase Count","Average Purchase Price", "T
otal Purchase Value", "Normalized Totals"]]
purch_analysis
```

Out[21]:

	Purchase Count	Average Purchase Price	Total Purchase Value	Normalized Totals
Gender				
Female	13	3.18	41.38	3.18
Male	64	2.88	184.60	3.08
Other / Non-Disclosed	1	2.12	2.12	2.12

In [22]:

```
#Bins for Age Demographics
bins = [0]
bin_value = ["<10"]
max_age = pymoli_data["Age"].max()

for x in range(2,int(max_age/5)+2):
    bins = bins + [(x*5)-1]

for x in range(1,len(bins)-2):
    bin_value = bin_value + [str(bins[x]+1)+"-"+str(bins[x+1])]

bin_value = bin_value + [str(max_age) + "+"]
#print(bins)
#print(bin_value)

#Age Demographics - need to group by SN.. add back in age, gender.. groupby ag
e group (count and percentage)
pymoli_data["Age Group"] = pd.cut(pymoli_data["Age"],bins,labels = bin_value)
```

In [23]:

```
#Purchasing Analysis (Age)
age_group = pymoli_data.groupby(["Age Group"])
age_count = age_group.size()
age_avg_price = round(age_group["Price"].mean(),2)
age_tot_value = age_group["Price"].sum()
age_norm_tot = round(age_tot_value/age_count,2)
age_dataframe = pd.DataFrame({"Purchase Count":age_count,
                             "Average Purchase Price": age_avg_price,
                             "Total Purchase Value": age_tot_value,
                             "Normalized Totals": age_norm_tot})

age_dataframe = age_dataframe[["Purchase Count","Average Purchase Price",
"Total Purchase Value", "Normalized Totals"]]
age_dataframe
```

Out[23]:

	Purchase Count	**Average Purchase Price**	**Total Purchase Value**	**Normalized Totals**
Age Group				
<10	5	2.76	13.82	2.76
10-14	3	2.99	8.96	2.99
15-19	11	2.76	30.41	2.76
20-24	36	3.02	108.89	3.02
25-29	9	2.90	26.11	2.90
30-34	7	1.98	13.89	1.98
35-39	6	3.56	21.37	3.56
40+	1	4.65	4.65	4.65

In [24]:

```
top_spend_group = pymoli_data.groupby("SN")
top_spend_count = top_spend_group.size()
top_spend_avg = round(top_spend_group["Price"].mean(),2)
top_spenders = top_spend_group["Price"].sum()
top_spenders_frame = pd.DataFrame({"Total Purchase Value": top_spenders})

#Identify top 5 spenders
top_spenders_sort = top_spenders_frame.sort_values(by="Total Purchase Value",
                                                   ascending = False)
top_spenders_filter = top_spenders_sort.iloc[0:5].reset_index()

#join spenders with other calculations
top_spend_func_frame = pd.DataFrame({"Purchase Count": top_spend_count,
                                     "Average Purchase Price": top_spend_avg})
.reset_index()

top_spenders_final = pd.merge(top_spenders_filter, top_spend_func_frame, on="S
N")
top_spenders_final
```

Out[24]:

	SN	Total Purchase Value	Average Purchase Price	Purchase Count
0	Sundaky74	7.41	3.70	2
1	Aidaira26	5.13	2.56	2
2	Eusty71	4.81	4.81	1
3	Chanirra64	4.78	4.78	1
4	Alarap40	4.71	4.71	1

In [25]:

```
top_items = pymoli_data.groupby("Item ID")
top_items_group2 = pymoli_data.groupby(["Item ID","Item Name","Price"])
top_items_count = top_items.size()
top_items_value = top_items_group2["Price"].sum()
top_items_frame = pd.DataFrame({"Purchase Count": top_items_count})
#Identify most popular items
top_items_sort = top_items_frame.sort_values(by="Purchase Count",
                                                  ascending = False).reset_
index()


end = top_items_sort["Purchase Count"].size

lastrow = 0

for row in range(4, end):
    if(top_items_sort.get_value(row,"Purchase Count") != top_items_sort.get
_value(row + 1,"Purchase Count")):
        lastrow = row
        break

if lastrow != 4:
    print("Note: As a result of ties, more than 5 items are the most popula
r. The tied values are printed")

top_items_filter = top_items_sort.iloc[0:lastrow+1].reset_index()

top_items_func_frame = pd.DataFrame({"Total Purchase Value": top_items_valu
e}).reset_index()

#merge tables
top_items_final = pd.merge(top_items_filter, top_items_func_frame, on="Item
 ID")

#fix column order and price name
top_items_final = top_items_final[["Item ID","Item Name","Purchase Count",
"Price","Total Purchase Value"]]
top_items_final = top_items_final.rename(columns={"Price":"Item Price"})
top_items_final
```

```
Note: As a result of ties, more than 5 items are the most popular. The tied v
alues are printed
```

Out[25]:

	Item ID	Item Name	Purchase Count	Item Price	Total Purchase Value
0	94	Mourning Blade	3	3.64	10.92
1	90	Betrayer	2	4.12	8.24
2	111	Misery's End	2	1.79	3.58
3	64	Fusion Pummel	2	2.42	4.84
4	154	Feral Katana	2	4.11	8.22
5	126	Exiled Mithril Longsword	2	1.08	2.16
6	117	Heartstriker, Legacy of the Light	2	4.71	9.42
7	60	Wolf	2	2.70	5.40
8	93	Apocalyptic Battlescythe	2	4.49	8.98
9	108	Extraction, Quickblade Of Trembling Hands	2	2.26	4.52
10	98	Deadline, Voice Of Subtlety	2	1.29	2.58
11	176	Relentless Iron Skewer	2	2.12	4.24
12	180	Stormcaller	2	2.77	5.54

In [26]:

```
#Most Profitable Items
top_values_frame = pd.DataFrame({"Total Purchase Value": top_items_value})
top_values_sort = top_values_frame.sort_values(by="Total Purchase Value",
                                                  ascending = False)
top_values_filter = top_values_sort.iloc[0:5].reset_index()
top_values_final = pd.merge(top_values_filter,top_items_sort, on="Item ID")

#rearrange columns and rename price column
top_values_final = top_values_final[["Item ID","Item Name","Purchase Count","P
rice","Total Purchase Value"]]
top_values_final = top_values_final.rename(columns={"Price":"Item Price"})
print()
top_values_final
```

Out[26]:

	Item ID	Item Name	Purchase Count	Item Price	Total Purchase Value
0	94	Mourning Blade	3	3.64	10.92
1	117	Heartstriker, Legacy of the Light	2	4.71	9.42
2	93	Apocalyptic Battlescythe	2	4.49	8.98
3	90	Betrayer	2	4.12	8.24
4	154	Feral Katana	2	4.11	8.22