In this project, I am using Pandas to perform basic analysis of a specific game's playerbase, using the Pandas and numpy libraries. With these, I was able to clean and sort data to uncover trends in player demographics, such as age and gender, and also to break down purchasing averages by these demographic breakdowns. I was also able to uncover the most purchased ingame items, and compare to see whether the top sellers were also the top sources of income.

In [424]:
```
# Dependencies and Setup
import pandas as pd
import numpy as np

# File to Load (Remember to Change These)
file_to_load = "Resources/purchase_data.csv"

# Read Purchasing File and store into Pandas data frame
purchase_df = pd.read_csv(file_to_load, header=[0])
display (purchase_df)
```

	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
...	...	...	...	...	...	...	...
775	775	Aethedru70	21	Female	60	Wolf	3.54
776	776	Iral74	21	Male	164	Exiled Doomblade	1.63
777	777	Yathecal72	20	Male	67	Celeste, Incarnation of the Corrupted	3.46
778	778	Sisur91	7	Male	92	Final Critic	4.19

Player Count

- Display the total number of players

In [425]:
```
player_count = len(pd.unique(purchase_df['SN']))
print("Total Player Count: ", player_count)
```

```
Total Player Count:  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 [426]:
```
unique_count = len(pd.unique(purchase_df['Item Name']))
total_purchase = sum(purchase_df['Price'])

#"currency" turns the raw number into proper currency format
total_currency = '${:,.2f}'.format(total_purchase)
average_price = purchase_df['Price'].mean()
avg_currency = '${:,.2f}'.format(average_price)
total_purchases = len(purchase_df['Purchase ID'])

purchasing_analysis = {'Unique Items':[unique_count], 'Average Purchase':[a
purch_analysis_df = pd.DataFrame(purchasing_analysis)
display(purch_analysis_df)
```

	Unique Items	Average Purchase	Total Purchase Value	Total Purchases
0	179	$3.05	$2,379.77	780

Gender Demographics

- Percentage and Count of Male Players
- Percentage and Count of Female Players
- Percentage and Count of Other / Non-Disclosed

In [427]:
```
#removing screen name duplicates gets accurate players to sort by gender
removed_dupes_df = purchase_df.drop_duplicates(subset=['SN'])
removed_dupes_genderlist = removed_dupes_df['Gender'].tolist()
male_players = removed_dupes_genderlist.count('Male')
male_percent = (male_players / len(removed_dupes_genderlist) * 100)
female_players = removed_dupes_genderlist.count('Female')
female_percent = (female_players / len(removed_dupes_genderlist) * 100)
nd_players = removed_dupes_genderlist.count('Other / Non-Disclosed')
nd_percent = (nd_players / len(removed_dupes_genderlist) * 100)

#save info in dictionary to use as dataframe
gender_demographic_data = {'Categories':['Male Players', 'Female Players','
gender_demographic_df = pd.DataFrame(gender_demographic_data)
gender_demographic_df['Percentage']=gender_demographic_df['Percentage'].ast
display(gender_demographic_df)
```

	Categories	Total	Percentage
0	Male Players	484	84.02777777777779%
1	Female Players	81	14.0625%
2	Other/Non-Disclosed	11	1.9097222222222223%

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 [428]:
```
#make a list for TOTAL PURCHASE
gender_purchase_list = purchase_df["Gender"].tolist()
male_purchases = gender_purchase_list.count('Male')
female_purchases = gender_purchase_list.count('Female')
nd_purchases = gender_purchase_list.count('Other / Non-Disclosed')
#use loc to create df's of price/gender
male_purchase_df = purchase_df.loc[purchase_df['Gender'] == 'Male', ['Gende
female_purchase_df = purchase_df.loc[purchase_df['Gender'] == 'Female', ['G
nd_purchase_df = purchase_df.loc[purchase_df['Gender'] == 'Other / Non-Disc
#calculate AVG PURCHASE
male_avg_purch = male_purchase_df['Price'].mean()
female_avg_purch = female_purchase_df['Price'].mean()
nd_avg_purch = nd_purchase_df['Price'].mean()
#calculate TOTAL PURCHASE VALUE BY GENDER
male_total_purchase = sum(male_purchase_df['Price'])
female_total_purchase = sum(female_purchase_df['Price'])
nd_total_purchase = sum(nd_purchase_df['Price'])
#calculate AVG PURCHASE PER PERSON BY GENDER
avg_perperson_male = male_total_purchase / male_players
avg_perperson_female = female_total_purchase / female_players
avg_perperson_nd = nd_total_purchase / nd_players
#format into currency
male_avg_purch = '${:,.2f}'.format(male_avg_purch)
female_avg_purch = '${:,.2f}'.format(female_avg_purch)
nd_avg_purch = '${:,.2f}'.format(nd_avg_purch)
male_total_purchase = '${:,.2f}'.format(male_total_purchase)
female_total_purchase = '${:,.2f}'.format(female_total_purchase)
nd_total_purchase = '${:,.2f}'.format(nd_total_purchase)
avg_perperson_male = '${:,.2f}'.format(avg_perperson_male)
avg_perperson_female = '${:,.2f}'.format(avg_perperson_female)
avg_perperson_nd = '${:,.2f}'.format(avg_perperson_nd)
#create dataframe from dictionary for results
gender_purchase_analysis = {'Gender': ['Male', 'Female', 'Other/Nondisclose
                            'Average Purchase Price':[male_avg_purch, femal
                            'Total Purchase Value': [male_total_purchase, f
                            'Avg Total Purchase Per Person': [avg_perperson
gender_purchase_analysis_df = pd.DataFrame(gender_purchase_analysis)
display(gender_purchase_analysis_df)
```

	Gender	Average Purchase Price	Total Purchase Value	Avg Total Purchase Per Person
0	Male	$3.02	$1,967.64	$4.07
1	Female	$3.20	$361.94	$4.47
2	Other/Nondisclosed	$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 [429]:
```
#create bins and bin labels
bins = [0, 9, 14, 19, 24, 29, 34, 39, float('inf')]
age_names = ('<10', '10-14', '15-19', '20-24', '25-29', '30-34', '35-39', '
#bin ages with purchase df
purchase_df["Age Analysis"] = pd.cut(purchase_df['Age'], bins, labels = age
#drop screenname duplicates for accurate count
age_count_df = purchase_df.drop_duplicates(subset='SN')
#save to list for counting
ages= age_count["Age Analysis"].tolist()
#save all values needed
under10 = ages.count('<10')
over10 = ages.count('10-14')
over15 = ages.count('15-19')
over20 = ages.count('20-24')
over25 = ages.count('25-29')
over30 = ages.count('30-34')
over35 = ages.count('35-39')
over40 = ages.count('40+')
under10percent = (under10 / len(ages)) * 100
over10percent = (over10 / len(ages)) * 100
over15percent = (over15 / len(ages)) * 100
over20percent = (over20 / len(ages)) * 100
over25percent = (over25 / len(ages)) * 100
over30percent = (over30 / len(ages)) * 100
over35percent = (over35 / len(ages)) * 100
over40percent = (over40 / len(ages)) * 100
#create dataframe to display information
player_age_analysis = {'Age Range':['<10', '10-14', '15-19', '20-24', '25-2
                       'Total Count':[under10, over10, over15, over20, ov
                       'Player Percentage':["{0:.2f}%".format(under10perc
player_age_analysis_df = pd.DataFrame(player_age_analysis)
display(player_age_analysis_df)
```

	Age Range	Total Count	Player Percentage
0	<10	17	2.95%
1	10-14	22	3.82%
2	15-19	107	18.58%
3	20-24	258	44.79%
4	25-29	77	13.37%
5	30-34	52	9.03%
6	35-39	31	5.38%
7	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 [430]:
```
#make a list for using count function
age_purchases = purchase_df["Age Analysis"].tolist()
punder10 = age_purchases.count('<10')
pover10 = age_purchases.count('10-14')
pover15 = age_purchases.count('15-19')
pover20 = age_purchases.count('20-24')
pover25 = age_purchases.count('25-29')
pover30 = age_purchases.count('30-35')
pover35 = age_purchases.count('35-39')
pover40 = age_purchases.count('40+')
#use loc to retrieve prices associated with age bins
punder10_purchase_df = purchase_df.loc[purchase_df['Age Analysis'] == '<10'
pover10_purchase_df = purchase_df.loc[purchase_df['Age Analysis'] == '10-14
pover15_purchase_df = purchase_df.loc[purchase_df['Age Analysis'] == '15-19
pover20_purchase_df = purchase_df.loc[purchase_df['Age Analysis'] == '20-24
pover25_purchase_df = purchase_df.loc[purchase_df['Age Analysis'] == '25-29
pover30_purchase_df = purchase_df.loc[purchase_df['Age Analysis'] == '30-34
pover35_purchase_df = purchase_df.loc[purchase_df['Age Analysis'] == '35-39
pover40_purchase_df = purchase_df.loc[purchase_df['Age Analysis'] == '40+',
#store average purchase values for display
avg_punder10 = "${0:.2f}".format(punder10_purchase_df['Price'].mean())
avg_pover10 = "${0:.2f}".format(pover10_purchase_df['Price'].mean())
avg_pover15 = "${0:.2f}".format(pover15_purchase_df['Price'].mean())
avg_pover20 = "${0:.2f}".format(pover20_purchase_df['Price'].mean())
avg_pover25 = "${0:.2f}".format(pover25_purchase_df['Price'].mean())
avg_pover30 = "${0:.2f}".format(pover30_purchase_df['Price'].mean())
avg_pover35 = "${0:.2f}".format(pover35_purchase_df['Price'].mean())
avg_pover40 = "${0:.2f}".format(pover40_purchase_df['Price'].mean())
#do the same for total
total_punder10 = "${0:.2f}".format(punder10_purchase_df['Price'].sum())
total_pover10 = "${0:.2f}".format(pover10_purchase_df['Price'].sum())
total_pover15 = "${0:.2f}".format(pover15_purchase_df['Price'].sum())
total_pover20 = "${0:.2f}".format(pover20_purchase_df['Price'].sum())
total_pover25 = "${0:.2f}".format(pover25_purchase_df['Price'].sum())
total_pover30 = "${0:.2f}".format(pover30_purchase_df['Price'].sum())
total_pover35 = "${0:.2f}".format(pover35_purchase_df['Price'].sum())
total_pover40 = "${0:.2f}".format(pover40_purchase_df['Price'].sum())
#going to use the df made above for average purchase per person, since it h
avg_punder10_perperson = "${0:.2f}".format(sum(punder10_purchase_df['Price'
avg_pover10_perperson = "${0:.2f}".format(sum(pover10_purchase_df['Price'])
avg_pover15_perperson = "${0:.2f}".format(sum(pover15_purchase_df['Price'])
avg_pover20_perperson = "${0:.2f}".format(sum(pover20_purchase_df['Price'])
avg_pover25_perperson = "${0:.2f}".format(sum(pover25_purchase_df['Price'])
avg_pover30_perperson = "${0:.2f}".format(sum(pover30_purchase_df['Price'])
avg_pover35_perperson = "${0:.2f}".format(sum(pover35_purchase_df['Price'])
avg_pover40_perperson = "${0:.2f}".format(sum(pover40_purchase_df['Price'])
#create dict for display
age_purchase_analysis = {'Age Range':['<10', '10-14', '15-19', '20-24', '25
                         'Purchase Count':[punder10, pover10, pover15, pover
                         'Average Purchase Price':[avg_punder10, avg_pover10
                         'Average Purchase Per Person':[avg_punder10_perpers
                         'Total Prices By Age':[total_punder10, total_pover1
#save to dataframe, display
age_purchase_analysis_df = pd.DataFrame(age_purchase_analysis)
display(age_purchase_analysis_df)
```

	Age Range	Purchase Count	Average Purchase Price	Average Purchase Per Person	Total Prices By Age
0	<10	23	$3.35	$4.54	$77.13
1	10-14	28	$2.96	$3.76	$82.78
2	15-19	136	$3.04	$3.86	$412.89
3	20-24	365	$3.05	$4.32	$1114.06
4	25-29	101	$2.90	$3.81	$293.00
5	30-34	0	$2.93	$4.12	$214.00
6	35-39	41	$3.60	$4.76	$147.67
7	40+	13	$2.94	$3.19	$38.24

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 [431]:

```
#count purchases in relation to screenname
name_count = purchase_df.groupby(['SN']).count()['Price']
name_count
#gather average purchase
name_avgprice = purchase_df.groupby(['SN']).mean()['Price']
name_avgprice
#gather total purchase value by name
name_totalpurch = purchase_df.groupby(['SN']).sum()['Price']
name_totalpurch
#make dataframe of these values, then sort
top_spenders = pd.DataFrame({'Purchase Count': name_count,
                             'Average Purchase': name_avgprice,
                             'Total Purchase Value': name_totalpurch,
                            })
top_spenders["Average Purchase"] = top_spenders["Average Purchase"].map("${
top_spenders = top_spenders.sort_values(by="Total Purchase Value", ascendin
#for some reason, styling the TPV above the sort value made this answer inc
top_spenders["Total Purchase Value"] = top_spenders["Total Purchase Value"]
#print head to 5 places
top_spenders.head(5)
```

Out[431]:

SN	Purchase Count	Average Purchase	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, average 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 [432]:
```
mostpopular_df = purchase_df[['Item ID', 'Item Name', 'Price']]

mostpopular_total = mostpopular_df.groupby(['Item Name']).sum()['Price']

mostpopular_average = mostpopular_df.groupby(['Item Name']).mean()['Price']

mostpopular_volume = mostpopular_df.groupby(['Item Name']).count()['Item ID

most_frequent_items_df = pd.DataFrame({'Purchase Count': mostpopular_volume
                            'Price of Item': mostpopular_average,
                            'Total Purchase Value': mostpopular_total,
                           })
most_frequent_items_df = most_frequent_items_df.sort_values(['Purchase Coun
most_frequent_items_df['Price of Item'] = most_frequent_items_df['Price of
most_frequent_items_df['Total Purchase Value'] = most_frequent_items_df['To
most_frequent_items_df.head(5)
```

Out[432]:

Item Name	Purchase Count	Price of Item	Total Purchase Value
Final Critic	13	$4.61	$59.99
Oathbreaker, Last Hope of the Breaking Storm	12	$4.23	$50.76
Nirvana	9	$4.90	$44.10
Fiery Glass Crusader	9	$4.58	$41.22
Extraction, Quickblade Of Trembling Hands	9	$3.53	$31.77

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 [434]:
```
#For some reason, formatting to currency messes things up.
mostprofit_df = purchase_df[['Item ID', 'Item Name', 'Price']]
#New df is untouched by formatting until declared later, fixing counts.
mostprofit_total = mostpopular_df.groupby(['Item Name']).sum()['Price']

mostprofit_average = mostpopular_df.groupby(['Item Name']).mean()['Price']

mostprofit_volume = mostpopular_df.groupby(['Item Name']).count()['Item ID'

most_profit_items_df = pd.DataFrame({'Purchase Count': mostprofit_volume,
                                     'Average Purchase': mostprofit_average,
                                     'Total Purchase Value': mostprofit_total,
                                    })
most_profit_items_df = most_profit_items_df.sort_values(by='Total Purchase
most_profit_items_df['Average Purchase'] = most_profit_items_df['Average Pu
most_profit_items_df['Total Purchase Value'] = most_profit_items_df['Total
most_profit_items_df.head(5)
```

Out[434]:

Item Name	Purchase Count	Average Purchase	Total Purchase Value
Final Critic	13	$4.61	$59.99
Oathbreaker, Last Hope of the Breaking Storm	12	$4.23	$50.76
Nirvana	9	$4.90	$44.10
Fiery Glass Crusader	9	$4.58	$41.22
Singed Scalpel	8	$4.35	$34.80