Note

- Instructions have been included for each segment. You do not have to follow them exactly, but they are included to help you think through the steps.

In [1]:
```
# Dependencies and Setup
import pandas as pd
import numpy as np
import pandas.io.formats.style

# 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_df = pd.read_csv(file_to_load)
```

Player Count

- Display the total number of players

In [2]:
```
#Print first 5 records of the csv data
purchase_data_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

In [3]:
```
#Display the total number of players
screen_name = purchase_data_df["SN"].value_counts().count()
#screen_name
total_players = pd.DataFrame({"Total Players": screen_name}, index=[0])

total_players
```

Out[3]:

	Total Players
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 [4]:
```
#Show the data types of the columns in the data
purchase_data_df.dtypes
```

Out[4]:
```
Purchase ID      int64
SN              object
Age              int64
Gender          object
Item ID          int64
Item Name       object
Price          float64
dtype: object
```

In [5]:
```
#Calculation to obtain the total number of records in the
total_count = purchase_data_df["SN"].count()

#Calculate the total revenue
total_revenue = purchase_data_df["Price"].sum()

#Calculate the number of unique items
total_unique = len(purchase_data_df["Item Name"].unique())

#Calculate the average price
avg_price = total_revenue / total_count
```

In [6]:
```
#Create a summary and give the columns different names by putting the values i
n a dictionary (Key/Value Pairs)
summary_df = pd.DataFrame({"Number of Unique Items": [total_unique],
                              "Average Price": avg_price,
                              "Number of Purchase": total_count,
                              "Total Revenue": total_revenue})

#Format 2 of the columns in the summary so they are showing as $
summary_df.style.format({'Average Price':"${:,.2f}",
                         'Total Revenue': '${:,.2f}'})
```

Out[6]:

	Number of Unique Items	Average Price	Number of Purchase	Total Revenue
0	179	$3.05	780	$2,379.77

Gender Demographics

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

In [7]:
```
#Group the Gender column so all same values are together
gender_counts_df = purchase_data_df.groupby("Gender")

#Calculate the unique screen names and get the number of gender for each so on
ly 1 value for every screen name is tallied
unique_SN_df = gender_counts_df["SN"].nunique()

#Calculate the gender % of players and format them
gender_percentage_df = round((unique_SN_df / screen_name)*100,2).map("{0:,.2f}
%".format)

#Create a summary for the gender demographics and sort highest to lowest by th
e total count
gender_summary_df = pd.DataFrame({"Total Count": unique_SN_df,
                               "Percentage of Players": gender_percentage_df})

#Gender demographics summary table sorted by total count
gender_summary_df.sort_values(by="Total Count", ascending=False)
```

Out[7]:

	Total Count	Percentage of Players
Gender		
Male	484	84.03%
Female	81	14.06%
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 [8]:
```
#Get purchase value count of all Genders even if duplicate purchases
purchase_count = purchase_data_df["Gender"].value_counts()

#Calculate average purchase price for each Gender regardless of duplicates
avg_purchase_df = round(purchase_data_df.groupby("Gender").Price.mean(),2)

#Calculate the total sum of the purchase price for each gender
total_gender_purch_df = round(purchase_data_df.groupby("Gender").Price.sum(),2
)

#Calculate the total average purchase price for each gender excluding duplicat
es
total_avg_purchase_df = round((total_gender_purch_df / unique_SN_df),2)

#Create a summary and give the columns different names by putting the values i
n a dictionary (Key/Value Pairs)
gender_purchasing_df = pd.DataFrame({"Purchase Count": purchase_count,
                              "Average Purchase Price": avg_purchase_df,
                              "Total Purchase Value": total_gender_purch_df,
                              "Avg Total Purchase per Person": total_avg_purch
ase_df})

#Sort the dataframe by purchase count
gender_purchasing_df = gender_purchasing_df.sort_values(by="Purchase Count", a
scending=False)

#Format 3 of the columns in the summary so they are showing as $
gender_purchasing_df.style.format({"Average Purchase Price":"${:,.2f}",
                                   "Total Purchase Value":"${:,.2f}",
                                   "Avg Total Purchase per Person":"${:,.2f}"
})
```

Out[8]:

	Purchase Count	Average Purchase Price	Total Purchase Value	Avg Total Purchase per Person
Male	652	$3.02	$1,967.64	$4.07
Female	113	$3.20	$361.94	$4.47
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 [9]:

```
# Establish bins for ages and create labels
demo_bins = [0, 9.90, 14.90, 19.90, 24.90, 29.90, 34.90, 39.90, 99999]
groups = ["<10", "10-14", "15-19", "20-24", "25-29", "30-34", "35-39", "40+"]

# slice the data using pd.cut and  Categorize the existing players based on ag
e_bins
purchase_data_df["Age Ranges"] = pd.cut(purchase_data_df["Age"],demo_bins, lab
els=groups)

#group dataframe by Age Ranges
age_group_df = purchase_data_df.groupby("Age Ranges")

#Calculate the unique screen names and get the number of gender for each so on
ly 1 value for every screen name is tallied
unique_age_df = age_group_df["SN"].nunique()

#Calculate the average player percentage for each age range
age_percentage_df = round((unique_age_df / screen_name)*100,2).map("{0:,.2f}%"
.format)

#Create the age range dataframe summary
age_summary_df = pd.DataFrame({"Total Count": unique_age_df,
                               "Percentage of Players": age_percentage_df})

age_summary_df
```

Out[9]:

	Total Count	Percentage of Players
Age Ranges		
<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 [10]:
```
purchase_data_df["Age Ranges"] = pd.cut(purchase_data_df["Age"],demo_bins, lab
els=groups)

#Group dataframe by Age Ranges
age_group_df = purchase_data_df.groupby("Age Ranges")

#Calculate the unique screen names and get the number of gender for each so on
ly 1 value for every screen name is tallied
unique_agegroup_df = age_group_df["SN"].nunique()

#Calculate average purchase price for each Age Range
age_avg_purchase = round(purchase_data_df.groupby("Age Ranges").Price.mean(),2
)

#Calculate the total sum of the purchase price for each Age Range
total_age_purch_df = round(purchase_data_df.groupby("Age Ranges").Price.sum(),
2)

#Calculate the total average purchase price for each Age Range excluding dupli
cates
total_age_avg_purchase = round((total_age_purch_df / unique_agegroup_df),2)

#Create the dataframe summary purchasing analysis by age range
age_purchasing_summary = pd.DataFrame({"Purchase Count": unique_agegroup_df,
                               "Average Purchase Price": age_avg_purchase,
                               "Total Purchase Value": total_age_purch_df,
                               "Avg Total Purchase per Person": total_age_avg_
purchase})

#Format the columns so the data has cleaner formatting
age_purchasing_summary.style.format({"Average Purchase Price":"${:,.2f}",
                                   "Total Purchase Value":"${:,.2f}",
                                   "Avg Total Purchase per Person":"${:,.2f}"
})
```

Out[10]:

	Purchase Count	Average Purchase Price	Total Purchase Value	Avg Total Purchase per Person
Age Ranges				
<10	17	$3.35	$77.13	$4.54
10-14	22	$2.96	$82.78	$3.76
15-19	107	$3.04	$412.89	$3.86
20-24	258	$3.05	$1,114.06	$4.32
25-29	77	$2.90	$293.00	$3.81
30-34	52	$2.93	$214.00	$4.12
35-39	31	$3.60	$147.67	$4.76
40+	12	$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 [11]:

```
#Group the dataframe by SN column
SN_df = purchase_data_df.groupby("SN")

#Count the unique values of the SN column
unique_SN_df = SN_df["SN"].count()

#Calculate the average price of the unique values in the SN column
age_avg_purchase = round(SN_df["Price"].mean(),2)

#Calculate the sum price of the unique values in the SN column
total_age_purch_df = round(SN_df["Price"].sum(),2)

#Top spenders summary dataframe
gender_purchasing_df = pd.DataFrame({"Purchase Count": unique_SN_df,
                              "Average Purchase Price": age_avg_purchase,
                              "Total Purchase Value": total_age_purch_df})

#Top spenders summary dataframe sorted and styled
gender_purchasing_df.sort_values(by="Total Purchase Value", ascending=False).h
ead().style.format({"Average Purchase Price":"${:,.2f}",
                                   "Total Purchase Value":"${:,.2f}"})
```

Out[11]:

	Purchase Count	Average Purchase Price	Total Purchase Value
SN			
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 [12]:
```
#Retrieve the Item ID, Item Name, and Item Price columns
popular_items_df = purchase_data_df[["Item ID", "Item Name","Price"]]

#Group by Item ID and Item Name
popular_items_grouped = popular_items_df.groupby(["Item ID", "Item Name"])

#Perform calculations to obtain purchase count, average item price, and total
 purchase value
most_popular_count = popular_items_grouped["Item Name"].count()
avg_most_popular = popular_items_grouped["Price"].mean()
total_most_popular = popular_items_grouped["Price"].sum()

#Most popular items summary dataframe
most_popular_summary = pd.DataFrame({"Purchase Count": most_popular_count,
                              "Average Purchase Price": avg_most_popular,
                              "Total Purchase Value": total_most_popular})

#Most popular items summary dataframe sorted by purchase count and formatted
most_popular_summary.sort_values(by="Purchase Count", ascending=False).head().
style.format({"Average Purchase Price":"${:,.2f}",
                                   "Total Purchase Value":"${:,.2f}"})
```

Out[12]:

		Purchase Count	Average Purchase Price	Total Purchase Value
Item ID	**Item Name**			
92	**Final Critic**	13	$4.61	$59.99
178	**Oathbreaker, Last Hope of the Breaking Storm**	12	$4.23	$50.76
145	**Fiery Glass Crusader**	9	$4.58	$41.22
132	**Persuasion**	9	$3.22	$28.99
108	**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 [13]:
```
#Most popular items summary dataframe sorted by total purchase value and forma
tted
most_popular_summary.sort_values(by="Total Purchase Value", ascending=False).h
ead().style.format({"Average Purchase Price":"${:,.2f}",
                                   "Total Purchase Value":"${:,.2f}"})
```

Out[13]:

		Purchase Count	Average Purchase Price	Total Purchase Value
Item ID	**Item Name**			
92	**Final Critic**	13	$4.61	$59.99
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
103	**Singed Scalpel**	8	$4.35	$34.80