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

# 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)

purchase_data.head()
```

Out[1]:

	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

Player Count

- Display the total number of players

In [2]:
```
# Get the list of unique players by SN and len of that list to get the count
total_players = len(purchase_data["SN"].unique())

#Display values
player_count = pd.DataFrame({"Total Players":[total_players]})
player_count
```

Out[2]:

	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 [3]:
```
# Calculating unique items
unique_items = len(purchase_data["Item Name"].unique())

# Calculating Average price by mean function
avg_price = purchase_data["Price"].mean()

# Calculating number of purchases
NumOfPurchase = len(purchase_data)

# Calculating Number of purchases
TotalRevenue = purchase_data["Price"].sum()

summary_df = pd.DataFrame({"Number of Unique Items" : [unique_items],
                           "Average Price" : [avg_price],
                           "Number of Purchases" : [NumOfPurchase],
                           "Total Revenue" : [TotalRevenue]
                           })

# Formating Average price & Total revnenue
summary_df["Average Price"] = summary_df["Average Price"].map("${:,.2f}".form
summary_df["Total Revenue"] = summary_df["Total Revenue"].map("${:,.2f}".form

# Printing the data frmae
summary_df
```

Out[3]:

	Number of Unique Items	Average Price	Number of Purchases	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 [4]:
```
# Group purchase data by Gender
gender_groups = purchase_data.groupby(["Gender"])

# Count the total of screen names "SN" by gender
total_count_gender = gender_groups["SN"].nunique()

#Total count by gender and divivde by total players
percent_player = (total_count_gender / total_players)*100

#Create data frame with obtained values
gender_df = pd.DataFrame({"Total Count" : total_count_gender,
               "Percentage of Players" : percent_player
              })

# Format the data frame with no index name
gender_df["Percentage of Players"] = gender_df["Percentage of Players"].map("
gender_df.index.name = None

# Sort the values according to total count
gender_df = gender_df.sort_values(by="Total Count", ascending=False)

# Print the dataframe
gender_df
```

Out[4]:

	Total Count	Percentage of Players
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 [5]:
```
# Getting purchase count purchase ID
purchase_count = gender_groups["Purchase ID"].count()

# Average purchase price
avg_price = gender_groups["Price"].mean()

# Total purchase value
total_value = gender_groups["Price"].sum()

# Average total purchase by person
Avg_total_purchase = total_value / total_count_gender

#Create data frame with obtained values
purchase_df = pd.DataFrame({"Purchase Count" : purchase_count,
                            "Average Purchase Price" : avg_price,
                            "Total Purchase Value" : total_value,
                            "Avg Total Purchase per Person" : Avg_total_purch
                           })

# Format the data frame
purchase_df["Average Purchase Price"] = purchase_df["Average Purchase Price"]
purchase_df["Total Purchase Value"] = purchase_df["Total Purchase Value"].map
purchase_df["Avg Total Purchase per Person"] = purchase_df["Avg Total Purchas
purchase_df
```

Out[5]:

	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

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 [6]:
```
#Get the min & max to create bins
min_val = purchase_data['Age'].min()
max_val = purchase_data['Age'].max()

min_val, max_val
```

Out[6]: (7, 45)

In [7]:
```
#Get the min & max to create bins

# Create the bins in which Data will be held
bins = [0, 9, 14, 19, 24, 29, 34, 39, 50]
group_labels = ['<10', '10-14', '15-19', '20-24', '25-29', '30-34', '35-39',

# categorized players by age bins and add a column as Age Group
purchase_data["Age Ranges"] = pd.cut(purchase_data["Age"], bins, labels=group
purchase_data.head()
```

Out[7]:

	Purchase ID	SN	Age	Gender	Item ID	Item Name	Price	Age Ranges
0	0	Lisim78	20	Male	108	Extraction, Quickblade Of Trembling Hands	3.53	20-24
1	1	Lisovynya38	40	Male	143	Frenzied Scimitar	1.56	40+
2	2	Ithergue48	24	Male	92	Final Critic	4.88	20-24
3	3	Chamassasya86	24	Male	100	Blindscythe	3.27	20-24
4	4	Iskosia90	23	Male	131	Fury	1.44	20-24

In [8]:
```
# Create a GroupBy object based upon "Age Ranges"
age_group = purchase_data.groupby("Age Ranges")

# getting count of unique players in age group
total_count = age_group["SN"].nunique()

# Getting percentage of players
percent_player_age = (total_count / total_players) * 100

# Create data frame with obtained values
age_df = pd.DataFrame({"Total Count" : total_count,
                       "Percentage of Players" : percent_player_age})

# Format the percentage of players
age_df["Percentage of Players"] = age_df["Percentage of Players"].map("{:,.2f

# Format data Frame with no index
age_df.index.name = None

# Display data frame
age_df
```

Out[8]:

	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%

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 [9]:
```
# Create a GroupBy object based upon Age Ranges
age_group1 = purchase_data.groupby("Age Ranges")

# Get the purchase count by age
purchase_count_by_age = age_group1["Purchase ID"].count()

# Get average price by age group
avg_price_by_age = age_group1["Price"].mean()

# Get the total price by age group
total_value_by_age = age_group1["Price"].sum()

# avg. purchase total per person by age group
avg_total_by_age = total_value_by_age / total_count

# Create data frmae with values obtained
purchase_age_df = pd.DataFrame({"Purchase Count" : purchase_count_by_age,
                                "Average Purchase Price" : avg_price_by_age,
                                "Total Purchase Value" : total_value_by_age,
                                "Avg Total Purchase per Person" : avg_total_by
                                })
# Format the columns
purchase_age_df["Average Purchase Price"] = purchase_age_df["Average Purchase
purchase_age_df["Total Purchase Value"] = purchase_age_df["Total Purchase Val
purchase_age_df["Avg Total Purchase per Person"] = purchase_age_df["Avg Total

# Display dataframe
purchase_age_df
```

Out[9]:

Age Ranges	Purchase Count	Average Purchase Price	Total Purchase Value	Avg Total Purchase per Person
<10	23	$3.35	$77.13	$4.54
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

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 [10]:
```
# Create a GroupBy object based upon SN
sn_grouped = purchase_data.groupby("SN")

# Get the purchase count by SN
top_purchase_count = sn_grouped["Purchase ID"].count()

# Get the total purchase by SN
top_total_purchase = sn_grouped["Price"].sum()

#Get the average purchase price by SN
top_avg_purchase = top_total_purchase / top_purchase_count

# Create a data frame from values obtained above
top_df = pd.DataFrame({"Purchase Count" : top_purchase_count,
                       "Average Purchase Price" : top_avg_purchase,
                       "Total Purchase Value" : top_total_purchase})

# Sort the values in descending order to get the top spenders
top_df = top_df.sort_values(by="Total Purchase Value", ascending=False)

# Format the values to display values till 2 decimal places & add $
top_df["Average Purchase Price"] = top_df["Average Purchase Price"].map("${:,
top_df["Total Purchase Value"] = top_df["Total Purchase Value"].map("${:,.2f}

# Display top 5 records (top 5 spenders) from data frame
top_df.head()
```

Out[10]:

SN	Purchase Count	Average Purchase Price	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 [11]:
```
# Extracted item id, item name & price to the new DF
item_df = purchase_data[["Item ID", "Item Name", "Price"]]

# Create a GroupBy object based upon Item ID & Item Name
grouped_item = item_df.groupby(["Item ID", "Item Name"])

# Get the purchase count grouped by Item ID & Item Name
purchase_count_item = grouped_item["Item ID"].count()

# Get the Total purchase value grouped by Item ID & Item Name
total_purchase_item = grouped_item["Price"].sum()

# Get the item price
item_price = total_purchase_item / purchase_count_item

# # Create df with the obtained values
popular_item_df = pd.DataFrame({"Purchase Count" : purchase_count_item,
                                "Item Price" : item_price,
                                "Total Purchase Value" : total_purchase_item}

# Sort the values in descending order to get the top spenders
popular_item_df = popular_item_df.sort_values(by="Purchase Count", ascending=

# Format the values to display values till 2 decimal places & add $
popular_item_df["Item Price"] = popular_item_df["Item Price"].map("${:,.2f}".
popular_item_df["Total Purchase Value"] = popular_item_df["Total Purchase Val

popular_item_df.head()
```

Out[11]:

Item ID	Item Name	Purchase Count	Item Price	Total Purchase Value
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 [12]:
```
profitable_items = popular_item_df.copy()

# remove '$' sign and convert back to float to sort
profitable_items["Total Purchase Value"] = profitable_items["Total Purchase V
profitable_items = profitable_items.sort_values("Total Purchase Value",ascend

# then add '$' and formatting back in
profitable_items["Total Purchase Value"] = profitable_items["Total Purchase V
profitable_items.head()
```

Out[12]:

Item ID	Item Name	Purchase Count	Item Price	Total Purchase Value
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