**Dependencies**

# Dependencies and Setup

import pandas as pd

import os

# File to Load (Remember to Change These)

#file\_to\_load = "Resources/purchase\_data.csv"

file\_to\_load = os.path.join('Resources','purchase\_data.csv')

# Read Purchasing File and store into Pandas data frame

purchase\_data = pd.read\_csv(file\_to\_load)

purchase\_data.head()

## Player Count

unique\_players = purchase\_data["SN"].unique()

unique\_players # Stores all unique players names

count = len(unique\_players) # count variable stores the count of unique players

#print (count)

unique\_df = pd.DataFrame({

"Total Players": [count]

})

unique\_df

## Purchasing Analysis (Total)

# using data functions

unique\_itm\_cnt = purchase\_data["Item ID"].nunique() # gives the count of unique items by counting "Item ID"

#unique\_itm\_cnt

dollar\_sign = '${:.2f}'

average\_price = purchase\_data["Price"].mean()

average\_price = dollar\_sign.format(average\_price) #formats average\_price by adding $ sign before and rounding to 2 digits

#average\_price

dollar\_sign = '${:.2f}'

total\_revenue = purchase\_data["Price"].sum()

total\_revenue = dollar\_sign.format(total\_revenue) #formats total\_revenue by adding $ sign before and rounding to 2 digits

#total\_revenue

purchase\_count = purchase\_data["Purchase ID"].count() #counts all purchase ids

#purchase\_count

#dataframe to store and display the calculations

purchase\_analysis\_df = pd.DataFrame({

"Number of Unique Items": [unique\_itm\_cnt],

"Average Price": [average\_price] ,

"Number of Purchases": [purchase\_count],

"Total Revenue": [total\_revenue]

})

purchase\_analysis\_df

## Gender Demographics

gender\_groupby = purchase\_data.groupby(['Gender']) #using groupby to get the count and percentage of gender categories

#gender\_groupby.count()

total\_count = gender\_groupby.nunique()["SN"] # counting unique values in all gender categories

total\_count

percentage\_gender = total\_count/count \*100 # count = 576 from Player count

percentage\_gender

# Create data frame to store and display

gender\_df = pd.DataFrame({

"Total Count": total\_count,

"Percentage of Players": percentage\_gender

})

#gender\_df

gender\_df.index.name = None

gender\_df["Percentage of Players"] = gender\_df["Percentage of Players"].map("{:.2f}%".format)

gender\_df = gender\_df.sort\_values(["Total Count"], ascending = False)

gender\_df

## Purchasing Analysis (Gender)

## gender\_grp = purchase\_data.groupby(['Gender']) # using groupby to do purchase analysis - by gender category

## gender\_grp.count()

## purchase\_count = gender\_grp["Purchase ID"].count() # using data functions to get values

## purchase\_count

## avg\_purchase\_price = gender\_grp["Price"].mean()

## avg\_purchase\_price

## tot\_purchase = gender\_grp["Price"].sum()

## tot\_purchase

## avg\_pur\_per\_person = tot\_purchase/total\_count

## #avg\_pur\_per\_person

## # create dataframe to store and display calculated results

## purchase\_analysis = pd.DataFrame({

## "Purchase Count": purchase\_count,

## "Average Purchase Price": avg\_purchase\_price,

## "Total Purchase Value": tot\_purchase,

## "Avg Total Purchase per Person": avg\_pur\_per\_person

## })

## # Map function to add $ signs and round values in required columns

## purchase\_analysis["Average Purchase Price"] = purchase\_analysis["Average Purchase Price"].map("${:.2f}".format)

## purchase\_analysis["Total Purchase Value"] = purchase\_analysis["Total Purchase Value"].map("${:.2f}".format)

## purchase\_analysis["Avg Total Purchase per Person"] = purchase\_analysis["Avg Total Purchase per Person"].map("${:.2f}".format)

## purchase\_analysis

## Age Demographics

## max\_age = purchase\_data["Age"].max()

## min\_age = purchase\_data["Age"].min()

## #min\_age

## #max\_age

## # creating a list of age ranges to analyse data in various age groups , to categorize data by age groups

## age\_ranges = ["<10","10-14","15-19","20-24","25-29","30-34","35-39","40+"]

## age\_bins = [0,9,14,19,24,29,34,39,120]

## # using pd.cut function for data binning

## purchase\_data["Age Ranges"] = pd.cut(purchase\_data["Age"],age\_bins,labels=age\_ranges)

## purchase\_data

## age\_groupby = purchase\_data.groupby(['Age Ranges']) # categrising data acc to age groups through group by

## age\_groupby.count()

## count\_by\_age = age\_groupby.nunique()["SN"] #unique player count in all age groups

## count\_by\_age

## percent\_by\_age = count\_by\_age/count\*100 # count = 576 from unique player count

## #percent\_by\_age

## #dataframe to store and display calculations

## age\_player = pd.DataFrame({

## "Total Count": count\_by\_age,

## "Percentage of Players": percent\_by\_age

## })

## age\_player["Percentage of Players"]=age\_player["Percentage of Players"].map("{:.2f}%".format)

## age\_player.index.name = None

## age\_player

## Purchasing Analysis (Age)

## age\_groupby = purchase\_data.groupby(['Age Ranges']) # categorizing by age range to for purchase analysis in all age groups

## age\_purchase\_count = age\_groupby["Purchase ID"].count() # counting purchases in all age groups

## age\_purchase\_count

## age\_avg\_price = age\_groupby["Price"].mean() # using data functions to find money spent age wise

## age\_avg\_price

## age\_tot\_price = age\_groupby["Price"].sum()

## age\_tot\_price

## age\_avg\_price\_per\_person = age\_tot\_price/count\_by\_age

## #age\_avg\_price\_per\_person

## #dataframe to store and display calculations

## age\_wise\_analysis = pd.DataFrame({

## "Purchase Count": age\_purchase\_count,

## "Average Purchase Price": age\_avg\_price,

## "Total Purchase Value": age\_tot\_price,

## "Avg Total Purchase per Person": age\_avg\_price\_per\_person

## })

## age\_wise\_analysis["Average Purchase Price"] = age\_wise\_analysis["Average Purchase Price"].map("${:.2f}".format)

## age\_wise\_analysis["Total Purchase Value"] = age\_wise\_analysis["Total Purchase Value"].map("${:.2f}".format)

## age\_wise\_analysis["Avg Total Purchase per Person"] = age\_wise\_analysis["Avg Total Purchase per Person"].map("${:.2f}".format)

## age\_wise\_analysis

## Top Spenders

## sn\_grp\_by = purchase\_data.groupby('SN') # categrizing data by player names

## sn\_purchase\_count = sn\_grp\_by['Purchase ID'].count() # using data functions to find each players purchase count , amount spent etc

## sn\_purchase\_count

## sn\_tot\_pur\_price = sn\_grp\_by['Price'].sum()

## sn\_tot\_pur\_price

## sn\_avg\_pur\_price = sn\_grp\_by['Price'].mean()

## #sn\_avg\_pur\_price

## sn\_data\_frame = pd.DataFrame({

## "Purchase Count": sn\_purchase\_count,

## "Average Purchase Price": sn\_avg\_pur\_price,

## "Total Purchase Value": sn\_tot\_pur\_price

## })

## sn\_data\_frame

## # to display values in sorted order as per Total purchase value - descending order

## sn\_data\_frame\_sorted = sn\_data\_frame.sort\_values("Total Purchase Value",ascending=False)

## # map function to formate requird columns with $ sign and rounding to 2 digits

## sn\_data\_frame\_sorted["Average Purchase Price"] = sn\_data\_frame\_sorted["Average Purchase Price"].map("${:.2f}".format)

## sn\_data\_frame\_sorted["Total Purchase Value"] = sn\_data\_frame\_sorted["Total Purchase Value"].map("${:.2f}".format)

## sn\_data\_frame\_sorted.head() # displays top 5 columns by default

## Most Popular Items

## Items\_df = purchase\_data[["Item ID","Item Name","Price"]] # getting sub data of whole data to analyse only sub data

## Items\_df

## item\_groupby = Items\_df.groupby(['Item ID','Item Name'])

## item\_groupby.count()

## item\_pur\_count = item\_groupby["Item ID"].count() # using data functions to get item wise purchase count and amount spent

## item\_pur\_count

## items\_pur\_val = item\_groupby["Price"].sum()

## items\_pur\_val

## one\_item\_price = items\_pur\_val/item\_pur\_count

## #one\_item\_price

## #data frame to store and display values

## item\_df = pd.DataFrame ({

## "Purchase Count": item\_pur\_count,

## "Item Price": one\_item\_price,

## "Total Purchase Value": items\_pur\_val

## })

## item\_df

## item\_df\_sorted = item\_df.sort\_values("Purchase Count", ascending = False) # sorting in descending order of purchase count

## #map functions to format required columns

## item\_df\_sorted["Item Price"] = item\_df\_sorted["Item Price"].map("${:.2f}".format)

## item\_df\_sorted["Total Purchase Value"] = item\_df\_sorted["Total Purchase Value"].map("${:.2f}".format)

## item\_df\_sorted.head() # displays top 5 columns by default

## Most Profitable Items

## # sorting as per Total Purchase Value - descending to know profitable items

## item\_df\_pur\_sort = item\_df.sort\_values("Total Purchase Value", ascending = False)

## #map function to add $ sign and rounding values in required columns

## item\_df\_pur\_sort["Item Price"] = item\_df\_pur\_sort["Item Price"].map("${:.2f}".format)

## item\_df\_pur\_sort["Total Purchase Value"] = item\_df\_pur\_sort["Total Purchase Value"].map("${:.2f}".format)

## item\_df\_pur\_sort.head() # displays top 5 columns by default