01.Importing Libraries

In [4]:
```
# Importing Libraries
import pandas as pd
import numpy as np
import os
```

02 Importing Data

In [7]:
```
df = pd.read_csv(r'C:\Users\Asus\Music\Instacart Basket Analysis\Data\Original Data/orders.csv', index_col = False)
```

In [9]:
```
# Telling Python to remember a main folder path
path = r'C:\Users\Asus\Music\Instacart Basket Analysis'
```

In [11]:
```
path
```

Out[11]:
```
'C:\\Users\\Asus\\Music\\Instacart Basket Analysis'
```

In [19]:
```
# Simplify the import function
df = pd.read_csv(os.path.join(path, 'Data', 'Original Data', 'orders.csv'), index_col = False)
```

03 Exploring Data with Pandas

In [22]:
```
df.head()
```

Out[22]:

	order_id	user_id	eval_set	order_number	order_dow	order_hour_of_day	days_since_prior_order
0	2539329	1	prior	1	2	8	NaN
1	2398795	1	prior	2	3	7	15.0
2	473747	1	prior	3	3	12	21.0
3	2254736	1	prior	4	4	7	29.0
4	431534	1	prior	5	4	15	28.0

In [24]:
```
df.head(30)
```

Out[24]:

	order_id	user_id	eval_set	order_number	order_dow	order_hour_of_day	days_since_prior_order
0	2539329	1	prior	1	2	8	NaN
1	2398795	1	prior	2	3	7	15.0
2	473747	1	prior	3	3	12	21.0
3	2254736	1	prior	4	4	7	29.0
4	431534	1	prior	5	4	15	28.0
5	3367565	1	prior	6	2	7	19.0
6	550135	1	prior	7	1	9	20.0
7	3108588	1	prior	8	1	14	14.0
8	2295261	1	prior	9	1	16	0.0
9	2550362	1	prior	10	4	8	30.0
10	1187899	1	train	11	4	8	14.0
11	2168274	2	prior	1	2	11	NaN
12	1501582	2	prior	2	5	10	10.0
13	1901567	2	prior	3	1	10	3.0
14	738281	2	prior	4	2	10	8.0
15	1673511	2	prior	5	3	11	8.0
16	1199898	2	prior	6	2	9	13.0
17	3194192	2	prior	7	2	12	14.0
18	788338	2	prior	8	1	15	27.0
19	1718559	2	prior	9	2	9	8.0
20	1447487	2	prior	10	1	11	6.0
21	1402090	2	prior	11	1	10	30.0
22	3186735	2	prior	12	1	9	28.0
23	3268552	2	prior	13	4	11	30.0
24	839880	2	prior	14	3	10	13.0
25	1492625	2	train	15	1	11	30.0
26	1374495	3	prior	1	1	14	NaN
27	444309	3	prior	2	3	19	9.0
28	3002854	3	prior	3	3	16	21.0
29	2037211	3	prior	4	2	18	20.0

In [26]:
```
df.tail()
```

Out[26]:

	order_id	user_id	eval_set	order_number	order_dow	order_hour_of_day	days_since_prior_order
3421078	2266710	206209	prior	10	5	18	29.0
3421079	1854736	206209	prior	11	4	10	30.0
3421080	626363	206209	prior	12	1	12	18.0
3421081	2977660	206209	prior	13	1	12	7.0
3421082	272231	206209	train	14	6	14	30.0

In [28]:
```
df.shape
```

Out[28]:
```
(3421083, 7)
```

In [30]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3421083 entries, 0 to 3421082
Data columns (total 7 columns):
 #   Column                  Dtype  
---  ------                  -----  
 0   order_id                int64  
 1   user_id                 int64  
 2   eval_set                object 
 3   order_number            int64  
 4   order_dow               int64  
 5   order_hour_of_day       int64  
 6   days_since_prior_order  float64
dtypes: float64(1), int64(5), object(1)
memory usage: 182.7+ MB
```

In [32]:
```
df.dtypes
```

Out[32]:
```
order_id                  int64
user_id                   int64
eval_set                 object
order_number              int64
order_dow                 int64
order_hour_of_day         int64
days_since_prior_order  float64
dtype: object
```

In [34]:
```
df.columns
```

Out[34]:
```
Index(['order_id', 'user_id', 'eval_set', 'order_number', 'order_dow',
       'order_hour_of_day', 'days_since_prior_order'],
      dtype='object')
```

In [36]:
```
df.describe()
```

Out[36]:

	order_id	user_id	order_number	order_dow	order_hour_of_day	days_since_prior_order
count	3.421083e+06	3.421083e+06	3.421083e+06	3.421083e+06	3.421083e+06	3.214874e+06
mean	1.710542e+06	1.029782e+05	1.715486e+01	2.776219e+00	1.345202e+01	1.111484e+01
std	9.875817e+05	5.953372e+04	1.773316e+01	2.046829e+00	4.226088e+00	9.206737e+00
min	1.000000e+00	1.000000e+00	1.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
25%	8.552715e+05	5.139400e+04	5.000000e+00	1.000000e+00	1.000000e+01	4.000000e+00
50%	1.710542e+06	1.026890e+05	1.100000e+01	3.000000e+00	1.300000e+01	7.000000e+00
75%	2.565812e+06	1.543850e+05	2.300000e+01	5.000000e+00	1.600000e+01	1.500000e+01
max	3.421083e+06	2.062090e+05	1.000000e+02	6.000000e+00	2.300000e+01	3.000000e+01

04. Importing datasets as a lists

Importing "orders.csv" omitting the "eval_set" column

In [40]:
```
# Creating a list of necessary columns
vars_list = ['order_id', 'user_id', 'order_number', 'order_dow', 'order_hour_of_day', 'days_since_prior_order']
```

In [42]:
```
# Import the "orders.csv" data set into the notebook using the os library and the vars_list shortcut, omitting the "eval_set" column
df = pd.read_csv(os.path.join(path, 'Data', 'Original Data', 'orders.csv'), usecols = vars_list)
```

In [44]:
```
# Print the first 5 rows of the data frame
df.head()
```

Out[44]:

	order_id	user_id	order_number	order_dow	order_hour_of_day	days_since_prior_order
0	2539329	1	1	2	8	NaN
1	2398795	1	2	3	7	15.0
2	473747	1	3	3	12	21.0
3	2254736	1	4	4	7	29.0
4	431534	1	5	4	15	28.0

Importing and exploring the "products.csv" dataframe

In [47]:
```
# Import the "products.csv" file into Jupyter
df_prods = pd.read_csv(os.path.join(path, 'Data', 'Original Data', 'products.csv'), index_col = False)
```

In [49]:
```
# Print the first 20 rows of the df_prods dataframe
df_prods.head(20)
```

Out[49]:

	product_id	product_name	aisle_id	department_id	prices
0	1	Chocolate Sandwich Cookies	61	19	5.8
1	2	All-Seasons Salt	104	13	9.3
2	3	Robust Golden Unsweetened Oolong Tea	94	7	4.5
3	4	Smart Ones Classic Favorites Mini Rigatoni Wit...	38	1	10.5
4	5	Green Chile Anytime Sauce	5	13	4.3
5	6	Dry Nose Oil	11	11	2.6
6	7	Pure Coconut Water With Orange	98	7	4.4
7	8	Cut Russet Potatoes Steam N' Mash	116	1	1.1
8	9	Light Strawberry Blueberry Yogurt	120	16	7.0
9	10	Sparkling Orange Juice & Prickly Pear Beverage	115	7	8.4
10	11	Peach Mango Juice	31	7	2.8
11	12	Chocolate Fudge Layer Cake	119	1	9.4
12	13	Saline Nasal Mist	11	11	1.1
13	14	Fresh Scent Dishwasher Cleaner	74	17	6.5
14	15	Overnight Diapers Size 6	56	18	11.2
15	16	Mint Chocolate Flavored Syrup	103	19	5.2
16	17	Rendered Duck Fat	35	12	17.1
17	18	Pizza for One Suprema Frozen Pizza	79	1	10.6
18	19	Gluten Free Quinoa Three Cheese & Mushroom Blend	63	9	12.0
19	20	Pomegranate Cranberry & Aloe Vera Enrich Drink	98	7	6.0

In [51]:
```
# Print the last 35 rows of the df_prods dataframe
df_prods.tail(35)
```

Out[51]:

	product_id	product_name	aisle_id	department_id	prices
49658	49654	Teriyaki Sauce, Sesame, Original	5	13	4.0
49659	49655	Apple Cider	98	7	10.7
49660	49656	Masada Kosher Pocket Bread	128	3	7.1
49661	49657	Cabernet Tomatoes	83	4	8.3
49662	49658	Brie with Herbs Foil Wedge	2	16	3.9
49663	49659	Organic Creamed Coconut	17	13	3.1
49664	49660	Professionals Sleek Shampoo	22	11	6.7
49665	49661	Porto	134	5	8.2
49666	49662	Bacon Cheddar Pretzel Pieces	107	19	3.6
49667	49663	Ultra Protein Power Crunch Peanut Butter N' Ho...	57	14	10.2
49668	49664	Lemon Cayenne Drinking Vinegar	100	21	13.7
49669	49665	Super Dark Coconut Ash & Banana Chocolate Bar	45	19	6.9
49670	49666	Ginger Snaps Snacking Cookies	61	19	5.2
49671	49667	Enchilada with Spanish Rice & Beans Meal	38	1	6.6
49672	49668	Apple Cinnamon Scented Candles	101	17	5.6
49673	49669	K Cup Dark Blend	100	21	4.7
49674	49670	Beef Summer Sausage	106	12	19.2
49675	49671	Milk Chocolate Drops	45	19	3.0
49676	49672	Cafe Mocha K-Cup Packs	26	7	6.5
49677	49673	Stone Baked Multi Grain Artisan Rolls	129	1	5.6
49678	49674	Frozen Greek Yogurt Bars Chocolate Chip	37	1	11.1
49679	49675	Cinnamon Dolce Keurig Brewed K Cups	26	7	14.0
49680	49676	Ultra Red Energy Drink	64	7	14.5
49681	49677	Thick & Chunky Sloppy Joe Sauce	59	15	8.9
49682	49678	Large Chicken & Cheese Taquitos	129	1	3.4
49683	49679	Famous Chocolate Wafers	61	19	6.0
49684	49680	All Natural Creamy Caesar Dressing	89	13	4.9
49685	49681	Spaghetti with Meatballs and Sauce Meal	38	1	6.9
49686	49682	California Limeade	98	7	4.3
49687	49683	Cucumber Kirby	83	4	13.2
49688	49684	Vodka, Triple Distilled, Twist of Vanilla	124	5	5.3
49689	49685	En Croute Roast Hazelnut Cranberry	42	1	3.1
49690	49686	Artisan Baguette	112	3	7.8
49691	49687	Smartblend Healthy Metabolism Dry Cat Food	41	8	4.7
49692	49688	Fresh Foaming Cleanser	73	11	13.5

In [53]:
```
# Print the names of the columns
df_prods.columns
```

Out[53]:
```
Index(['product_id', 'product_name', 'aisle_id', 'department_id', 'prices'], dtype='object')
```

In [55]:
```
# Print the number of rows and columns
df_prods.shape
```

Out[55]:
```
(49693, 5)
```

In [57]:
```
df_prods.dtypes
```

Out[57]:
```
product_id       int64
product_name    object
aisle_id         int64
department_id    int64
prices         float64
dtype: object
```

The data type of the "department_id" column is integer64

In [60]:
```
df_prods.describe()
```

Out[60]:

	product_id	aisle_id	department_id	prices
count	49693.000000	49693.000000	49693.000000	49693.000000
mean	24844.345139	67.770249	11.728433	9.994136
std	14343.717401	38.316774	5.850282	453.519696
min	1.000000	1.000000	1.000000	1.000000
25%	12423.000000	35.000000	7.000000	4.100000
50%	24845.000000	69.000000	13.000000	7.100000
75%	37265.000000	100.000000	17.000000	11.200000

max	49688.000000	134.000000	21.000000	99999.000000

In [ ]: