Final Project - Analyzing Sales Data

Date: 11 December 2022

Author: Tanapohn ekkanjanagorn

Course: `Pandas Foundation`

```
import numpy as np
import pandas as pd
```

```
# import data
import pandas as pd
df = pd.read_csv("sample-store.csv")
```

```
# preview top 5 rows
df.head()
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country/Region	City
0	1	CA-2019-152156	11/8/2019	11/11/2019	Second Class	CG-12520	Claire Gute	Consumer	United States	Hend
1	2	CA-2019-152156	11/8/2019	11/11/2019	Second Class	CG-12520	Claire Gute	Consumer	United States	Hend
2	3	CA-2019-138688	6/12/2019	6/16/2019	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Ange
3	4	US-2018-108966	10/11/2018	10/18/2018	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Laude
4	5	US-2018-108966	10/11/2018	10/18/2018	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Laude

5 rows × 21 columns

```
# shape of dataframe
df.shape
```

```
(9994, 21)
```

```
# see data frame information using .info()
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 21 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Row ID          9994 non-null   int64
 1   Order ID        9994 non-null   object
 2   Order Date      9994 non-null   object
 3   Ship Date       9994 non-null   object
 4   Ship Mode       9994 non-null   object
 5   Customer ID     9994 non-null   object
 6   Customer Name   9994 non-null   object
 7   Segment         9994 non-null   object
 8   Country/Region  9994 non-null   object
```

```
 9   City            9994 non-null   object
 10  State           9994 non-null   object
 11  Postal Code     9983 non-null   float64
 12  Region          9994 non-null   object
 13  Product ID      9994 non-null   object
```

```
# change column name to lower case and replaec space & hyphen (-) with undersc
df.columns = df.columns.str.lower()
df.columns = df.columns.str.replace(' ', '_')
df.columns = df.columns.str.replace('-', '_')
df.head(3)
```

	row_id	order_id	order_date	ship_date	ship_mode	customer_id	customer_name	segment	country/reg
0	1	CA-2019-152156	11/8/2019	11/11/2019	Second Class	CG-12520	Claire Gute	Consumer	United State
1	2	CA-2019-152156	11/8/2019	11/11/2019	Second Class	CG-12520	Claire Gute	Consumer	United State
2	3	CA-2019-138688	6/12/2019	6/16/2019	Second Class	DV-13045	Darrin Van Huff	Corporate	United State

3 rows × 21 columns

We can use `pd.to_datetime()` function to convert columns 'Order Date' and 'Ship Date' to datetime.

```
# example of pd.to_datetime() function
pd.to_datetime(df['order_date'].head(), format='%m/%d/%Y')
```

```
0   2019-11-08
1   2019-11-08
2   2019-06-12
3   2018-10-11
4   2018-10-11
Name: order_date, dtype: datetime64[ns]
```

```
# TODO - convert order date and ship date to datetime in the original datafram
df['order_date'] = pd.to_datetime(df['order_date'], format= '%m/%d/%Y')
df['ship_date'] = pd.to_datetime(df['ship_date'], format= '%m/%d/%Y')
df.head()
```

	row_id	order_id	order_date	ship_date	ship_mode	customer_id	customer_name	segment	country/regio
0	1	CA-2019-152156	2019-11-08	2019-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States
1	2	CA-2019-152156	2019-11-08	2019-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States
2	3	CA-2019-138688	2019-06-12	2019-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States
3	4	US-2018-108966	2018-10-11	2018-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States
4	5	US-2018-108966	2018-10-11	2018-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States

5 rows × 21 columns

```
# TODO - count nan in postal code column
df.isna().sum()
```

```
row_id             0
order_id           0
order_date         0
ship_date          0
ship_mode          0
customer_id        0
customer_name      0
segment            0
country/region     0
city               0
state              0
postal_code       11
region             0
product_id         0
category           0
sub_category       0
product_name       0
sales              0
quantity           0
discount           0
profit             0
dtype: int64
```

```
# TODO - filter rows with missing values
df[df['postal_code'].isna()]
```

	row_id	order_id	order_date	ship_date	ship_mode	customer_id	customer_name	segment	country/re
2234	2235	CA-2020-104066	2020-12-05	2020-12-10	Standard Class	QJ-19255	Quincy Jones	Corporate	United Sta
5274	5275	CA-2018-162887	2018-11-07	2018-11-09	Second Class	SV-20785	Stewart Visinsky	Consumer	United Sta
8798	8799	US-2019-150140	2019-04-06	2019-04-10	Standard Class	VM-21685	Valerie Mitchum	Home Office	United Sta
9146	9147	US-2019-165505	2019-01-23	2019-01-27	Standard Class	CB-12535	Claudia Bergmann	Corporate	United Sta
9147	9148	US-2019-165505	2019-01-23	2019-01-27	Standard Class	CB-12535	Claudia Bergmann	Corporate	United Sta
9148	9149	US-2019-165505	2019-01-23	2019-01-27	Standard Class	CB-12535	Claudia Bergmann	Corporate	United Sta
9386	9387	US-2020-127292	2020-01-19	2020-01-23	Standard Class	RM-19375	Raymond Messe	Consumer	United Sta
9387	9388	US-2020-127292	2020-01-19	2020-01-23	Standard Class	RM-19375	Raymond Messe	Consumer	United Sta
9388	9389	US-2020-127292	2020-01-19	2020-01-23	Standard Class	RM-19375	Raymond Messe	Consumer	United Sta
9389	9390	US-2020-127292	2020-01-19	2020-01-23	Standard Class	RM-19375	Raymond Messe	Consumer	United Sta
9741	9742	CA-2018-117086	2018-11-08	2018-11-12	Standard Class	QJ-19255	Quincy Jones	Corporate	United Sta

11 rows × 21 columns

```
# drop missing value
df2 = df.dropna()
```

```
df2.columns
```

```
Index(['row_id', 'order_id', 'order_date', 'ship_date', 'ship_mode',
       'customer_id', 'customer_name', 'segment', 'country/region', 'city',
```

```
       'state', 'postal_code', 'region', 'product_id', 'category',
       'sub_category', 'product_name', 'sales', 'quantity', 'discount',
       'profit'],
      dtype='object')
```

```
df2.state.unique()
```

```
array(['Kentucky', 'California', 'Florida', 'North Carolina',
       'Washington', 'Texas', 'Wisconsin', 'Utah', 'Nebraska',
       'Pennsylvania', 'Illinois', 'Minnesota', 'Michigan', 'Delaware',
       'Indiana', 'New York', 'Arizona', 'Virginia', 'Tennessee',
       'Alabama', 'South Carolina', 'Oregon', 'Colorado', 'Iowa', 'Ohio',
       'Missouri', 'Oklahoma', 'New Mexico', 'Louisiana', 'Connecticut',
       'New Jersey', 'Massachusetts', 'Georgia', 'Nevada', 'Rhode Island',
       'Mississippi', 'Arkansas', 'Montana', 'New Hampshire', 'Maryland',
       'District of Columbia', 'Kansas', 'Maine', 'South Dakota', 'Idaho',
       'North Dakota', 'Wyoming', 'West Virginia'], dtype=object)
```

```
# TODO - Explore this dataset on your owns, ask your own questions
# 1. How many items and how much sales are there sold in 2019 by Category - Su
q1 = df2.groupby(['category','sub_category'])[['quantity','sales']].sum()
q1
```

		quantity	sales
category	sub_category		
Furniture	Bookcases	863	110475.0963
	Chairs	2353	327733.9030
	Furnishings	3563	91705.1640
	Tables	1241	206965.5320
Office Supplies	Appliances	1726	106989.2210
	Art	2994	27110.7520
	Binders	5974	203412.7330
	Envelopes	905	16474.3620
	Fasteners	914	3024.2800
	Labels	1400	12486.3120
	Paper	5173	78387.0060
	Storage	3145	222279.3180
	Supplies	647	46673.5380
Technology	Accessories	2967	167075.3080
	Copiers	234	149528.0300
	Machines	440	189238.6310
	Phones	3284	328712.3040

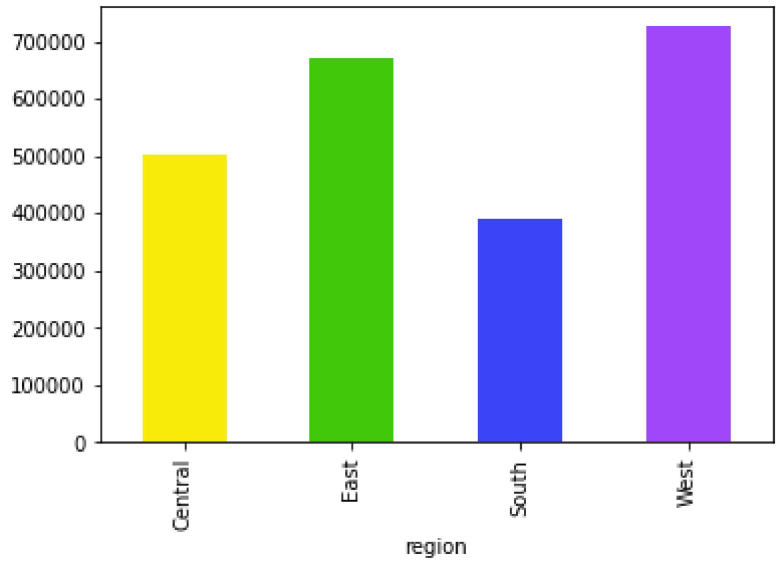
```
# 2. Max, Min , Sum and Average sales value, shipment with first class mode
q2 = df2[df2['ship_mode'] == 'First Class']['sales'].agg(['max','min','sum', '
q2
```

```
max       13999.960000
min           0.984000
sum      351428.422900
mean        228.497024
count      1538.000000
Name: sales, dtype: float64
```

```
# 3.Bar chart of sale values category by region
q3 = df2.groupby('region')['sales'].sum().plot(kind = 'bar', color = ['#F7EC09
q3
```

```
<AxesSubplot:xlabel='region'>
```

Download



Data Analysis Part

Answer 10 below questions to get credit from this course. Write `pandas` code to find answers.

```
# TODO 01 - how many columns, rows in this dataset
df.shape

# Ans: 9994 columns, 21 rows.
```

```
(9994, 21)
```

```
# TODO 02 - is there any missing values?, if there is, which colunm? how many
df.isna().sum()

# Ans: There are 11 missing values in 'Postal code' column.
```

```
row_id             0
order_id           0
order_date         0
ship_date          0
ship_mode          0
customer_id        0
customer_name      0
segment            0
country/region     0
city               0
state              0
postal_code       11
region             0
product_id         0
category           0
sub_category       0
product_name       0
sales              0
quantity           0
discount           0
profit             0
dtype: int64
```

```python
# TODO 03 - your friend ask for `California` data, filter it and export csv fo
# select California
df3 = df2[df2['state'] == 'California']

# reset index
df3 = df3.reset_index(drop= True)

# export data
df3.to_csv('store_california.csv')
df3
```

	row_id	order_id	order_date	ship_date	ship_mode	customer_id	customer_name	segment	country/re
0	3	CA-2019-138688	2019-06-12	2019-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United Sta
1	6	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United Sta
2	7	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United Sta
3	8	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United Sta
4	9	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United Sta
...	...	...	...	...	...	...	...	...	...
1996	9987	CA-2019-125794	2019-09-29	2019-10-03	Standard Class	ML-17410	Maris LaWare	Consumer	United Sta
1997	9991	CA-2020-121258	2020-02-26	2020-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United Sta
1998	9992	CA-2020-121258	2020-02-26	2020-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United Sta
1999	9993	CA-2020-121258	2020-02-26	2020-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United Sta
2000	9994	CA-2020-119914	2020-05-04	2020-05-09	Second Class	CC-12220	Chris Cortes	Consumer	United Sta

2001 rows × 21 columns

```
# TODO 04 - your friend ask for all order data in `California` and `Texas` in

# Filter data between two dates & filter state
df4 = df2.query("order_date >= '2017-01-01' and order_date < '2018-01-01' \
                & state == ['California', 'Texas']")

# export data
df4.to_csv('store_2017_California&Texas.csv')
df4
```

	row_id	order_id	order_date	ship_date	ship_mode	customer_id	customer_name	segment	country/re
5	6	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United Sta
6	7	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United Sta
7	8	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United Sta
8	9	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United Sta
9	10	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United Sta
...	...	...	...	...	...	...	...	...	...
9885	9886	CA-2017-112291	2017-04-03	2017-04-08	Standard Class	KE-16420	Katrina Edelman	Corporate	United Sta
9903	9904	CA-2017-122609	2017-11-12	2017-11-18	Standard Class	DP-13000	Darren Powers	Consumer	United Sta
9904	9905	CA-2017-122609	2017-11-12	2017-11-18	Standard Class	DP-13000	Darren Powers	Consumer	United Sta
9942	9943	CA-2017-143371	2017-12-28	2018-01-03	Standard Class	MD-17350	Maribeth Dona	Consumer	United Sta
9943	9944	CA-2017-143371	2017-12-28	2018-01-03	Standard Class	MD-17350	Maribeth Dona	Consumer	United Sta

632 rows × 21 columns

```
# TODO 05 - how much total sales, average sales, and standard deviation of sal

# select order date in 2017
df5 = df2.query("order_date >= '2017-01-01' and order_date < '2018-01-01'")

# aggregate
df5['sales'].agg(['sum', 'mean', 'std'])
```

```
sum     484247.498100
mean       242.974159
std        754.053357
Name: sales, dtype: float64
```

```
# TODO 06 - which Segment has the highest profit in 2018
# select date range
df6 = df2.query("order_date >= '2018-01-01' and order_date < '2019-01-01'")

# group $ sum data
df6 = df6.groupby('segment')['profit'].sum()

# sort values (descending)
df6.sort_values(ascending=False)
```

```
segment
Consumer       28281.3665
Corporate      19675.1978
Home Office    12470.1124
Name: profit, dtype: float64
```

```
# TODO 07 - which top 5 States have the least total sales between 15 April 201
# select date range
df7 = df2.query("order_date >= '2019-04-15' and order_date < '2020-01-01'")

# group & sum data
df7 = df7.groupby('state')['sales'].sum()

# sort values (ascending) and select first 5 rows
df7.sort_values(ascending=True).head(5)
```

```
state
New Hampshire            49.05
New Mexico               64.08
District of Columbia    117.07
Louisiana               249.80
South Carolina          502.48
Name: sales, dtype: float64
```

```python
# TODO 08 - what is the proportion of total sales (%) in West + Central in 201
# select date range
df8 = df2.query("order_date >= '2019-01-01' and order_date < '2020-01-01'")

df8_sum_sale = df8['sales'].sum()

# group & sum data (set index = false)
df8 = df8.groupby('region', as_index=False)['sales'].sum()

# append column percent
df8["percent"] = round(df8['sales'] / df8['sales'].sum() * 100 , 2)
df8
```

	region	sales	percent
0	Central	147429.3760	24.32
1	East	177718.7620	29.31
2	South	93610.2235	15.44
3	West	187480.1765	30.93

```python
# Ans 08 : total sales (%) in West + Central in 2019
df8.query(' region == ["Central" , "West"]')['percent'].sum()
```

```
55.25
```

```python
# TODO 09 - find top 10 popular products in terms of number of orders vs. tota

# select date range
df9 = df2.query("order_date >= '2019-01-01' and order_date < '2021-01-01'")

# create rank column
rank = pd.Series(['1','2','3','4','5','6','7','8','9','10'], name = 'Ranks')

# find top 10
    # 1) find total number of order & sales group by product name
    # 2) sort value by descending
    # 3) select first 10 rows ()
    # 4) rename colunm head
    # 5) reset index

# top 10 number of orders
top10orders = df9.groupby('product_name', as_index=False)['quantity'].sum()\
                .sort_values(by='quantity', ascending=False)\
                .head(10)\
                .rename(columns = {'product_name':'top10_orders'})\
                .reset_index()


# top 10 sales value
top10sales = df9.groupby('product_name', as_index=False)['sales'].sum()\
                .sort_values(by='sales', ascending=False)\
                .head(10)\
                .rename(columns = {'product_name':'top10_sales'})\
                .reset_index()

# concatenate dataframes
top10 = pd.concat([rank, top10orders['top10_orders'], top10sales['top10_sales'
top10
```

	Ranks	top10_orders	top10_sales
0	1	Staples	Canon imageCLASS 2200 Advanced Copier
1	2	Easy-staple paper	Hewlett Packard LaserJet 3310 Copier
2	3	Staple envelope	3D Systems Cube Printer, 2nd Generation, Magenta
3	4	Staples in misc. colors	GBC Ibimaster 500 Manual ProClick Binding System
4	5	Chromcraft Round Conference Tables	GBC DocuBind TL300 Electric Binding System
5	6	Storex Dura Pro Binders	GBC DocuBind P400 Electric Binding System
6	7	Situations Contoured Folding Chairs, 4/Set	Samsung Galaxy Mega 6.3
7	8	Wilson Jones Clip & Carry Folder Binder Tool f...	HON 5400 Series Task Chairs for Big and Tall
8	9	Avery Non-Stick Binders	Martin Yale Chadless Opener Electric Letter Op...
9	10	Wilson Jones Turn Tabs Binder Tool for Ring Bi...	Global Troy Executive Leather Low-Back Tilter

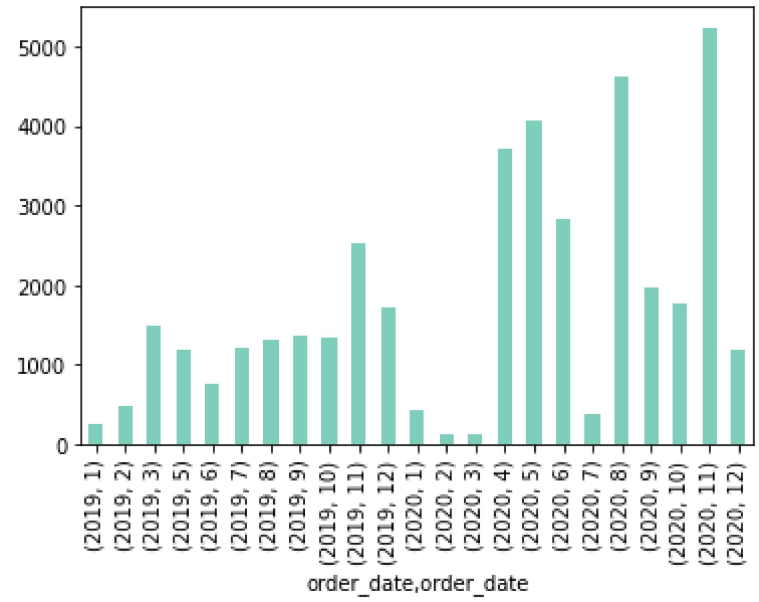
```
# TODO 10 - plot at least 2 plots, any plot you think interesting :)

# Graph 1 : Monthly sales value in Florida between 2019-2020.

 # filter order date from 2019 to 2020 & select state
 # sum total sales monthly
dfs = df2.query("order_date >= '2019-01-01' and order_date < '2021-01-01' & st
        .groupby([df2.order_date.dt.year, df2.order_date.dt.month])['sales'].s

 # plot bar chart
dfs.plot(kind = 'bar', color = '#7fcdbb');
```

Download



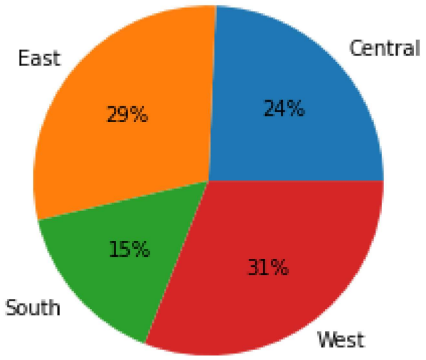
```
# Graph 2 : The proportion of total sales (%) in 2019 by region

import matplotlib.pyplot as plt

 # plot pie chart from df (with labeled)
plt.pie(df8['percent'], labels=df8['region'], autopct='%1.0f%%')
```

```
([<matplotlib.patches.Wedge at 0x7f415dc449a0>,
  <matplotlib.patches.Wedge at 0x7f415dc44fa0>,
  <matplotlib.patches.Wedge at 0x7f415dc52700>,
  <matplotlib.patches.Wedge at 0x7f415dc52e20>],
 [Text(0.7942550881228482, 0.7610248714667391, 'Central'),
  Text(-0.8466621491155206, 0.7024968541671445, 'East'),
  Text(-0.8319414225887488, -0.7196342608443606, 'South'),
  Text(0.6202909331466325, -0.908426748976537, 'West')],
 [Text(0.4332300480670081, 0.41510447534549405, '24%'),
  Text(-0.46170662679028396, 0.3831801022729878, '29%'),
  Text(-0.45378623050295386, -0.3925277786423784, '15%'),
  Text(0.33834050898907225, -0.4955054994417474, '31%')])
```

Download




```
# TODO Bonus - use np.where() to create new column in dataframe to help you an
 # Q : Find the orders in california that made profit more than 200 in 2020.

# create year column
df['year'] = df['order_date'].dt.year

# filter year 2019
df[df['year'] == 2019]

# Summary
df['goal'] = np.where(df['profit'] > 20, True,False)
df
```

	row_id	order_id	order_date	ship_date	ship_mode	customer_id	customer_name	segment	country/re
0	1	CA-2019-152156	2019-11-08	2019-11-11	Second Class	CG-12520	Claire Gute	Consumer	United Sta
1	2	CA-2019-152156	2019-11-08	2019-11-11	Second Class	CG-12520	Claire Gute	Consumer	United Sta
2	3	CA-2019-138688	2019-06-12	2019-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United Sta
3	4	US-2018-108966	2018-10-11	2018-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United Sta
4	5	US-2018-108966	2018-10-11	2018-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United Sta
...	...	...	...	...	...	...	...	...	...
9989	9990	CA-2017-110422	2017-01-21	2017-01-23	Second Class	TB-21400	Tom Boeckenhauer	Consumer	United Sta
9990	9991	CA-2020-121258	2020-02-26	2020-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United Sta
9991	9992	CA-2020-121258	2020-02-26	2020-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United Sta
9992	9993	CA-2020-121258	2020-02-26	2020-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United Sta
9993	9994	CA-2020-119914	2020-05-04	2020-05-09	Second Class	CC-12220	Chris Cortes	Consumer	United Sta

9994 rows × 23 columns