In [ ]:
```
1. convert all 3 sheets to csv
2. create tables and load data for all 3 sheets
```

In [15]:
```
#pip install mysql-connector-python
```

In [1]:
```
import pandas as pd
import mysql.connector as con
```

Que: 1,2. a: Creating dataframe and loading data into pandas

1. Importing all sheets as df in pandas
2. Converting these df to csv files for loading data into MySQL using csvkit.

In [2]:
```
df_o = pd.read_excel('Superstore_USA.xlsx', sheet_name= 'Orders' )
```

In [3]:
```
df_o
```

Out[3]:

	Row ID	Order Priority	Discount	Unit Price	Shipping Cost	Customer ID	Customer Name	Ship Mode	Customer Segment	Product Category	...	Region	State or Province	City	Po C
0	18606	Not Specified	0.01	2.88	0.50	2	Janice Fletcher	Regular Air	Corporate	Office Supplies	...	Central	Illinois	Addison	60
1	20847	High	0.01	2.84	0.93	3	Bonnie Potter	Express Air	Corporate	Office Supplies	...	West	Washington	Anacortes	98
2	23086	Not Specified	0.03	6.68	6.15	3	Bonnie Potter	Express Air	Corporate	Office Supplies	...	West	Washington	Anacortes	98
3	23087	Not Specified	0.01	5.68	3.60	3	Bonnie Potter	Regular Air	Corporate	Office Supplies	...	West	Washington	Anacortes	98
4	23088	Not Specified	0.00	205.99	2.50	3	Bonnie Potter	Express Air	Corporate	Technology	...	West	Washington	Anacortes	98
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9421	20275	Critical	0.06	35.89	14.72	3402	Frederick Cole	Regular Air	Consumer	Office Supplies	...	East	West Virginia	Charleston	25
9422	20276	Critical	0.00	3.34	7.49	3402	Frederick Cole	Regular Air	Consumer	Office Supplies	...	East	West Virginia	Charleston	25
9423	24491	Not Specified	0.08	550.98	45.70	3402	Frederick Cole	Delivery Truck	Consumer	Furniture	...	East	West Virginia	Charleston	25
9424	25914	High	0.10	105.98	13.99	3403	Tammy Buckley	Express Air	Consumer	Furniture	...	West	Wyoming	Cheyenne	82
9425	24492	Not Specified	0.09	7.78	2.50	3403	Tammy Buckley	Express Air	Consumer	Office Supplies	...	West	Wyoming	Cheyenne	82

9426 rows × 24 columns

In [5]:
```
df_o.to_csv('superstore_orders.csv', index= False)
```

In [4]:
```
df_r = pd.read_excel('Superstore_USA.xlsx', sheet_name= 'Returns' )
```

In [5]:
```
df_r
```

Out[5]:

	Order ID	Status
0	65	Returned
1	612	Returned
2	614	Returned
3	678	Returned
4	710	Returned
...	...	...
1629	182681	Returned
1630	182683	Returned
1631	182750	Returned

1632	182781	Returned
1633	182906	Returned

1634 rows × 2 columns

In [21]:
```
df_r.to_csv('superstore_returns.csv', index= False)
```

In [6]:
```
df_u = pd.read_excel('Superstore_USA.xlsx' , sheet_name= 'Users')
```

In [7]:
```
df_u
```

Out[7]:

	Region	Manager
0	Central	Chris
1	East	Erin
2	South	Sam
3	West	William

In [22]:
```
df_u.to_csv('superstore_users.csv', index= False)
```

Que: 1,2 b Creating database, tables in MySQL and loading data into these tables using csvkit.

In [33]:
```
mydb = con.connect(host= 'localhost', user = 'root', passwd='123456')
cursor = mydb.cursor()
print(mydb)
```

```
<mysql.connector.connection_cext.CMySQLConnection object at 0x0000020D652A2490>
```

In [18]:
```
cursor.execute("create database superstore_USA")
```

Using `csvkit` in `anaconda prompt` to create table and load data into table.

1. First requirement of `mysqlclient` and `csvkit` was already fulfilled as i have installed these for fitbit task.
2. By using **csvsql --db mysql+mysqldb://root:123456@localhost:3306/superstore_USA --tables orders --insert superstore_orders.csv** created table `orders` and loaded data into it from csv file `superstore_orders.csv`
3. By using **csvsql --db mysql+mysqldb://root:123456@localhost:3306/superstore_USA --tables returns --insert superstore_returns.csv** created table `returns` and loaded data into it from csv file `superstore_returns.csv`
4. By using **svsql --db mysql+mysqldb://root:123456@localhost:3306/superstore_USA --tables users --insert superstore_users.csv** created table `users` and loaded data into it from csv file `superstore_users.csv`

Que: 3. Find out how many return that we have recieved and with a product id

In [8]:
```
df_o.columns
```

Out[8]:
```
Index(['Row ID', 'Order Priority', 'Discount', 'Unit Price', 'Shipping Cost',
       'Customer ID', 'Customer Name', 'Ship Mode', 'Customer Segment',
       'Product Category', 'Product Sub-Category', 'Product Container',
       'Product Name', 'Product Base Margin', 'Region', 'State or Province',
       'City', 'Postal Code', 'Order Date', 'Ship Date', 'Profit',
       'Quantity ordered new', 'Sales', 'Order ID'],
      dtype='object')
```

In [9]:
```
df_r.columns
```

```
Index(['Order ID', 'Status'], dtype='object')
```

```
Out[9]: Index(['Order ID', 'Status'], dtype='object')
```

```
In [10]: df1 = pd.merge(df_o, df_r)
```

since order and return table both doesnt have a product_id column so using product name for the calculations

There is total 98 returns in total, this number is low because there is very less order id in order and returns table that match.

```
In [11]: df1.groupby('Product Name')['Status'].value_counts().sum()
```

```
Out[11]: 98
```

No of returns for each product is as below:

```
In [12]: df1.groupby('Product Name')['Status'].value_counts()
```

```
Out[12]:
Product Name                                   Status
#10 White Business Envelopes,4 1/8 x 9 1/2     Returned    1
12 Colored Short Pencils                       Returned    1
232                                            Returned    1
600 Series Flip                                Returned    1
6160                                           Returned    1
                                                          ..
Xerox 197                                      Returned    2
Xerox 1980                                     Returned    1
Xerox 1983                                     Returned    2
Xerox 210                                      Returned    1
Zoom V.92 V.44 PCI Internal Controllerless FaxModem  Returned    1
Name: Status, Length: 94, dtype: int64
```

```
In [13]: # not part of solution
         df1[df1['Status'] == 'Returned'][['Product Name', 'Order ID','Status']]
```

Out[13]:

	Product Name	Order ID	Status
0	Dixon My First Ticonderoga Pencil, #2	9895	Returned
1	Avery 493	13959	Returned
2	EcoTones® Memo Sheets	13959	Returned
3	Newell 35	36038	Returned
4	Staples SlimLine Pencil Sharpener	39490	Returned
...	...	...	...
93	Snap-A-Way® Black Print Carbonless Ruled Speed...	7107	Returned
94	Recycled Steel Personal File for Standard File...	7107	Returned
95	Xerox 1980	42823	Returned
96	TDK 4.7GB DVD+RW	13638	Returned
97	SAFCO Folding Chair Trolley	47109	Returned

98 rows × 3 columns

Que: 4 Try to join order and return data both in sql and pandas

Ans: for pandas join of order and return table is in previous question and same is given below

`df1 = pd.merge(df_o, df_r)`

Join orders and returns using MySQL

```
In [20]: query1 = """
                 select *
                 from superstore_usa.orders
                 join superstore_usa.returns
                     using(`Order ID`);
                 """
```

```
cursor.execute(query1)

for i in cursor.fetchone() :
    print(i)
```

```
9895
1359
Low
0.05
5.85
2.27
21
Tony Wilkins Winters
Regular Air
Small Business
Office Supplies
Pens & Art Supplies
Wrap Bag
Dixon My First Ticonderoga Pencil, #2
0.56
East
New York
New York City
10012
2011-04-20
2011-04-24
-6.820000000000000000
9
54.79
Returned
```

Que 5 Try to find out how many unique customer that we have

In [14]:
```
df_o.groupby('Customer Name')['Customer Name'].unique()
```

Out[14]:
```
Customer Name
Aaron Davies Bruce          [Aaron Davies Bruce]
Aaron Day                            [Aaron Day]
Aaron Dillon                      [Aaron Dillon]
Aaron Fuller Davidson    [Aaron Fuller Davidson]
Aaron Riggs                        [Aaron Riggs]
                                  ...
Zachary House                    [Zachary House]
Zachary Maynard                [Zachary Maynard]
Zachary Potter                  [Zachary Potter]
Zachary Wu                          [Zachary Wu]
Zachary Yu                          [Zachary Yu]
Name: Customer Name, Length: 2703, dtype: object
```

Que: 6 Try to find out in how many regions we are selling a product and who is a manager for a respective region

In [23]:
```
df2 = pd.merge(df_o, df_u)
```

In [25]:
```
df2.head(2)
```

Out[25]:

	Row ID	Order Priority	Discount	Unit Price	Shipping Cost	Customer ID	Customer Name	Ship Mode	Customer Segment	Product Category	...	State or Province	City	Postal Code	Order Date	Ship Date
0	18606	Not Specified	0.01	2.88	0.50	2	Janice Fletcher	Regular Air	Corporate	Office Supplies	...	Illinois	Addison	60101	2012-05-28	2012-05-3
1	24844	Medium	0.09	78.69	19.99	14	Gwendolyn F Tyson	Regular Air	Small Business	Furniture	...	Minnesota	Prior Lake	55372	2010-05-12	2010-05-1

2 rows × 25 columns

In [45]:
```
print("Total no of managers is: ",len(df2.groupby('Region')['Manager'].unique()))
```

```
Total no of managers is:  4
```

In [48]:
```
print("Total No of regions in which we are selling is: ", len(df2.groupby('Region')['Region'].unique()))
```
```
Total No of regions in which we are selling is:  4
```

Que: 7 . Find out how many different different shipment mode that we have and what is a percentage usablity of all the shipment mode with respect to dataset

In [51]:
```
print("Total No of shipment modes is: ",len(df_o.groupby('Ship Mode')['Ship Mode'].unique()))
```
```
Total No of shipment modes is:  3
```

In [65]:
```
shipment_modes_count = list(zip(df_o.groupby('Ship Mode')['Ship Mode'].unique(), df_o.groupby('Ship Mode')['Ship

total_shipments = 0
shipment_mode_list = []
for i,j in shipment_modes_count :
    total_shipments += j
    shipment_mode_list.append(j)

total_shipments
```
Out[65]:
```
9426
```

In [70]:
```
delivery_Truck = (shipment_mode_list[0] / total_shipments)*100

print("Percentage of usability of Delivery Truck for shipment is: ", round(delivery_Truck,2))
```
```
Percentage of usability of Delivery Truck for shipment is:  13.61
```

In [71]:
```
express_air = (shipment_mode_list[1] / total_shipments)*100

print("Percentage of usability of Express air for shipment is: ", round(express_air,2))
```
```
Percentage of usability of Express air for shipment is:  11.74
```

In [72]:
```
regular_air = (shipment_mode_list[2] / total_shipments)*100

print("Percentage of usability of Regular air for shipment is: ", round(regular_air,2))
```
```
Percentage of usability of Regular air for shipment is:  74.64
```

Method 2

In [46]:
```
shipment_mode_Individual_total = df_o['Ship Mode'].value_counts()
shipment_mode_Individual_total
```
Out[46]:
```
Regular Air       7036
Delivery Truck    1283
Express Air       1107
Name: Ship Mode, dtype: int64
```

In [41]:
```
df_shipment_mode = pd.DataFrame(shipment_mode_Individual_total)
total_shipment_across_all_mode = df_shipment_mode.sum()
total_shipment_across_all_mode
```
Out[41]:
```
Ship Mode    9426
dtype: int64
```

In [49]:
```
((df_shipment_mode[['Ship Mode']])/ total_shipment_across_all_mode ) * 100
```

Out[49]:

	Ship Mode
Regular Air	74.644600
Delivery Truck	13.611288
Express Air	11.744112

Que: 8 Create a new coulmn and try to find our a diffrence between order date and shipment date

In [51]:
```
df_o['No_of_days'] = df_o['Ship Date'] - df_o['Order Date']
```

In [53]:
```
print("Maximum difference between ship_date and order_date is: ",df_o['No_of_days'].dt.components['days'].max())
```
```
Maximum difference between ship_date and order_date is:  92
```

In [54]:
```
print("Minimum difference between ship_date and order_date is: ",df_o['No_of_days'].dt.components['days'].min())
```
```
Minimum difference between ship_date and order_date is:  0
```

In [58]:
```
print("Mean of difference between ship_date and order_date is: ", round(df_o['No_of_days'].dt.components['days'].
```
```
Mean of difference between ship_date and order_date is:  2.029
```

Que: 9 based on question number 8 find out for which order id we have shipment duration more than 10 days

Assumption: since shipment duration is not given directly or indirectly hence considering time taken for shipment to reach customer after ship date to be constant for all orders, and considering Pseudo shipment duration to be the difference of ship date and order date i.e. No_of_days_int

In [69]:
```
df_o['No_of_days_int'] = df_o['No_of_days'].dt.components['days']
```

In [76]:
```
df_o[df_o['No_of_days_int'] > 10]['Order ID']
```

Out[76]:
```
643     87215
1548    86318
1549    86318
1678    87957
1679    87957
1680    87957
1697    19556
1698    19556
1699    19556
2515    86177
5548    88223
5673    88352
5859    87572
5881    91294
8607    86434
8609    86436
8610    86436
8973    87300
8982    19841
8983    19841
8993    19841
8996    87300
8997    87300
Name: Order ID, dtype: int64
```

In [77]:
```
print("No. of Order ID where shipment Duration is more than 10 days is: ", len(df_o[df_o['No_of_days'].dt.compone
```
```
No. of Order ID where shipment Duration is more than 10 days is:  23
```

Que: 10 . Try to find out a list of a returned order which shipment duration was more then 15 days and find out that region manager as well

Assumption: since shipment duration is not given directly or indirectly hence considering time taken for shipment to reach customer after ship date to be constant for all orders, and considering Pseudo shipment duration to be the difference of ship date and order date i.e. No_of_days_int

In [101…
```
df3 = pd.merge(df_o, df_u)
```

In [105…
```
df3[df3['No_of_days_int'] > 15][['Order ID', 'Manager']]
```

Out[105…

	Order ID	Manager
189	87215	Chris
3476	87957	William
3478	87957	William
3495	19556	William
3497	19556	William
3826	86177	William
4476	91294	William
6638	88352	Erin
7314	19841	Erin
7315	19841	Erin
7325	19841	Erin
7328	87300	Erin
7329	87300	Erin
8435	87572	Sam
9184	86434	Sam
9186	86436	Sam
9187	86436	Sam
9307	87300	Sam

Que: 11 . Group by region and find out which region is more profitable

In [107…
```
df_o.groupby(['Region'])[['Region','Profit']].sum()
```

Out[107…

	Profit
Region	
Central	519825.567067
East	377566.186045
South	104201.192420
West	310849.453897

Que: 12 Try to find out overall in which country we are giving more discount

Assumption Since the data is of USA only hence instead of country considering State or Province

In [109…
```
sorted(zip(df_o.groupby('State or Province')['State or Province'].unique(), df_o.groupby('State or Province')['Di
```

Out[109…
```
(array(['California'], dtype=object), 52.28)
```

Que: 13 . Give me a list of unique postal code

In [119…
```
unique_postal_code = df_o['Postal Code'].unique()
unique_postal_code
```

Out[119…
```
array([60101, 98221, 91776, ..., 61832, 62521, 26554], dtype=int64)
```

In [120…
```
len(unique_postal_code)
```

Out[120…
```
1697
```

Que: 14 . which customer segement is more profitalble find it out .

In [121…
```
df_o.groupby('Customer Segment')[['Customer Segment','Profit']].sum()
```

Out[121…

	Profit
Customer Segment	
Consumer	206559.625348
Corporate	505538.627783
Home Office	283869.553814
Small Business	316474.592482

Que: 15 Try to find out the 10th most loss making product catagory .

**Since only three product category is there so instead of product category using Product Sub-Category

In [122…
```
df_o.groupby('Product Category')[['Product Category','Profit']].sum()
```

Out[122…

	Profit
Product Category	
Furniture	177354.298188
Office Supplies	451990.216492
Technology	683097.884748

In [171…
```
sorted(zip(df_o.groupby('Product Sub-Category')['Product Sub-Category'].unique(), df_o.groupby('Product Sub-Categ
```

Out[171…
```
(array(['Computer Peripherals'], dtype=object), 87917.8425126)
```

Que: 16 . Try to find out 10 top product with highest margins

sale_margin = profit / sales

In [174…
```
df_o['Sale_margin'] = df_o['Profit'] / df_o['Sales']
```

In [180…
```
sorted(zip(df_o.groupby('Product Name')['Product Name'].unique(), df_o.groupby('Product Name')['Sale_margin'].sum
```

Out[180…
```
[(array(['Avery White Multi-Purpose Labels'], dtype=object),
  999.4917946970588),
 (array(['Rediform S.O.S. Phone Message Books'], dtype=object),
  756.9390799807127),
 (array(['Avery 506'], dtype=object), 209.36706212242635),
 (array(['Telephone Message Books with Fax/Mobile Section, 4 1/4" x 6"'],
        dtype=object),
  179.6019489865885),
 (array(['Avery 501'], dtype=object), 136.76074214276562),
 (array(['Avery 51'], dtype=object), 118.79854476537459),
 (array(['Plymouth Boxed Rubber Bands by Plymouth'], dtype=object),
  104.85428178095958),
 (array(['Xerox 1981'], dtype=object), 99.14039824272862),
 (array(['Staples 4 Outlet Surge Protector'], dtype=object),
  95.92155969684188),
 (array(['GBC Plastic Binding Combs'], dtype=object), 90.95879457291079),
 (array(['Staples #10 Laser & Inkjet Envelopes, 4 1/8" x 9 1/2", 100/Box'],
        dtype=object),
                89.41789504359595)]
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js