Midterm Skills Exam (2nd Sem 2024)

Instructions: For this part of the exam, answer the given questions in a jupyter notebook. All of the questions must be answered locally, through a brief explanation and Python code. Students are not allowed to look at any external references.

1. Define an ETL pipeline. In your own words, explain the purpose of each component: Extract, Transform, and Load.

An ETL pipeline is a method to clean or filter a dataset from inputs that are irrelevant to the topic. The components of ETL pipeline are:

- Extract, it is to get the dataset by running it into the program. It is the initial step before the dataset is cleaned.
- Transform, it is to clean the data from inputs that are irrelevant to the topic, such as dropping duplicated table and removing entries that are "N/A".
- Load, it is to run cleaned or filtered dataset, or in short the output from the process of transformation.

2. Define an ETL pipeline. In your own words, explain the purpose of each component: Extract, Transform, and Load.

An ETL pipeline is a method to clean or filter a dataset from inputs that are irrelevant to the topic. The components of ETL pipeline are:

- Extract, it is to get the dataset by running it into the program. It is the initial step before the dataset is cleaned.
- Transform, it is to clean the data from inputs that are irrelevant to the topic, such as dropping duplicated table and removing entries that are "N/A".
- Load, it is to run cleaned or filtered dataset, or in short the output from the process of transformation.

3. Identify at least two potential data quality issues that might be present in the provided fake data.

In [1]:
```
import pandas as pd
```

In [2]:
```
sales = pd.read_csv("sales_data_raw.csv")
sales = pd.DataFrame(sales)
customers = pd.read_csv("customers_data.json")
customers = pd.DataFrame(customers)
```

In [3]:
```
# There are no N/As.
sales.isna().any()
```

Out[3]:
```
TransactionID      False
CustomerID         False
TransactionDate    False
Product            False
Quantity           False
Price              False
Discount           False
dtype: bool
```

In [4]:
```
# There are no duplicates.
sales.duplicated().any()
```

Out[4]:
```
False
```

In [5]:
```
# The TransactionDate is in wrong data type, it must be in datatime data type.
sales.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   TransactionID    10 non-null     object 
 1   CustomerID       10 non-null     object 
 2   TransactionDate  10 non-null     object 
 3   Product          10 non-null     object 
 4   Quantity         10 non-null     int64  
 5   Price            10 non-null     int64  
 6   Discount         10 non-null     float64
dtypes: float64(1), int64(2), object(4)
memory usage: 692.0+ bytes
```

In [6]:
```
# The TransactionDate is in string form (object), it must be converted to datatime
sales["TransactionDate"] = sales["TransactionDate"].apply(pd.to_datetime)
sales.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype         
---  ------           --------------  -----         
 0   TransactionID    10 non-null     object        
 1   CustomerID       10 non-null     object        
 2   TransactionDate  10 non-null     datetime64[ns]
 3   Product          10 non-null     object        
 4   Quantity         10 non-null     int64         
 5   Price            10 non-null     int64         
 6   Discount         10 non-null     float64       
dtypes: datetime64[ns](1), float64(1), int64(2), object(3)
memory usage: 692.0+ bytes
```

In [7]:
```
# There is no proper index value
sales
```

Out[7]:

	TransactionID	CustomerID	TransactionDate	Product	Quantity	Price	Discount
0	T001	C001	2023-01-01	Widget	2	10	0.00
1	T002	C002	2023-01-05	Gadget	1	20	0.10
2	T003	C003	2023-01-07	Widget	3	10	0.00
3	T004	C002	2023-01-10	Gizmo	5	15	0.05
4	T005	C001	2023-01-12	Widget	1	10	0.00
5	T006	C004	2023-01-15	Gadget	2	20	0.20
6	T007	C005	2023-01-18	Widget	4	10	0.00
7	T008	C002	2023-01-20	Gizmo	3	15	0.10
8	T009	C003	2023-01-22	Widget	5	10	0.00
9	T010	C005	2023-01-25	Gadget	3	20	0.15

In [8]:
```
# The customers table is not in proper table format, It needs to be pivoted.
customers
```

Out[8]:

				[
{"CustomerID": "C001"	**"Name": "Alice"**	**"JoinDate": "2022-12-01"}**	NaN
{"CustomerID": "C002"	**"Name": "Bob"**	**"JoinDate": "2022-11-15"}**	NaN
{"CustomerID": "C003"	**"Name": "Charlie"**	**"JoinDate": "2023-01-05"}**	NaN
{"CustomerID": "C004"	**"Name": "Diana"**	**"JoinDate": "2023-01-10"}**	NaN
{"CustomerID": "C005"	**"Name": "Evan"**	**"JoinDate": "2023-01-20"}**	NaN
]	**NaN**	**NaN**	NaN

4. How would you transform the TransactionDate in the sales data and the JoinDate in the customer data into proper datetime objects using Pandas?

In [9]:
```
# To transfrom the TransactionDate to datetime using pandas, "pd.to_datetime(column
pd.to_datetime(sales["TransactionDate"])
sales.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   TransactionID    10 non-null     object
 1   CustomerID       10 non-null     object
 2   TransactionDate  10 non-null     datetime64[ns]
 3   Product          10 non-null     object
 4   Quantity         10 non-null     int64
 5   Price            10 non-null     int64
 6   Discount         10 non-null     float64
dtypes: datetime64[ns](1), float64(1), int64(2), object(3)
memory usage: 692.0+ bytes
```

In [10]:
```
sales
```

Out[10]:

	TransactionID	CustomerID	TransactionDate	Product	Quantity	Price	Discount
0	T001	C001	2023-01-01	Widget	2	10	0.00
1	T002	C002	2023-01-05	Gadget	1	20	0.10
2	T003	C003	2023-01-07	Widget	3	10	0.00
3	T004	C002	2023-01-10	Gizmo	5	15	0.05
4	T005	C001	2023-01-12	Widget	1	10	0.00
5	T006	C004	2023-01-15	Gadget	2	20	0.20
6	T007	C005	2023-01-18	Widget	4	10	0.00
7	T008	C002	2023-01-20	Gizmo	3	15	0.10
8	T009	C003	2023-01-22	Widget	5	10	0.00
9	T010	C005	2023-01-25	Gadget	3	20	0.15

Final Transaction Amount = (Quantity × Price) × (1 – Discount)

5. Write a function to calculate the final transaction amount given the columns Quantity, Price, and Discount.

In [11]:
```
# Use simple arithmethic computation of each columns as instructed and that is "(Qu
sales["Final Transaction Amount"] = (sales["Quantity"]*sales["Price"]) * (1 - sales
sales
```

Out[11]:

	TransactionID	CustomerID	TransactionDate	Product	Quantity	Price	Discount	Transa Am
0	T001	C001	2023-01-01	Widget	2	10	0.00	
1	T002	C002	2023-01-05	Gadget	1	20	0.10	
2	T003	C003	2023-01-07	Widget	3	10	0.00	
3	T004	C002	2023-01-10	Gizmo	5	15	0.05	
4	T005	C001	2023-01-12	Widget	1	10	0.00	
5	T006	C004	2023-01-15	Gadget	2	20	0.20	
6	T007	C005	2023-01-18	Widget	4	10	0.00	
7	T008	C002	2023-01-20	Gizmo	3	15	0.10	
8	T009	C003	2023-01-22	Widget	5	10	0.00	
9	T010	C005	2023-01-25	Gadget	3	20	0.15	

6. Explain how you would join the sales data with the customer data. Which column is the appropriate key to use?

In [12]:
```
combined = sales.concat([customers])
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_8588\1492115985.py in ?()
----> 1 combined = sales.concat([customers])

C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\generic.py in ?(self, name)
   6295             and name not in self._accessors
   6296             and self._info_axis._can_hold_identifiers_and_holds_name(name)
   6297         ):
   6298             return self[name]
-> 6299         return object.__getattribute__(self, name)

AttributeError: 'DataFrame' object has no attribute 'concat'
```

7. What method in Pandas would you use to identify and remove duplicate rows in the sales data?

In [13]:
```
# With the ".any()" function, it determines whether there are data that are duplica
sales.duplicated().any() # It is False, because there are no duplicated datas
```

Out[13]:
```
False
```

In [14]:
```
# to drop or remove duplicate rows in the sales data, use ".drop_duplicates()"
sales.drop_duplicates() # There were no rows dropped or removed, because there are
```

Out[14]:

	TransactionID	CustomerID	TransactionDate	Product	Quantity	Price	Discount	Transa Am
0	T001	C001	2023-01-01	Widget	2	10	0.00	
1	T002	C002	2023-01-05	Gadget	1	20	0.10	
2	T003	C003	2023-01-07	Widget	3	10	0.00	
3	T004	C002	2023-01-10	Gizmo	5	15	0.05	
4	T005	C001	2023-01-12	Widget	1	10	0.00	
5	T006	C004	2023-01-15	Gadget	2	20	0.20	
6	T007	C005	2023-01-18	Widget	4	10	0.00	
7	T008	C002	2023-01-20	Gizmo	3	15	0.10	
8	T009	C003	2023-01-22	Widget	5	10	0.00	
9	T010	C005	2023-01-25	Gadget	3	20	0.15	

8. After transforming the data, list two different methods you might use to load the data into a target system, including any relevant libraries or functions.

9. Are there other transformations that are necessary to perform on the dataset that were not included so far? List down and perform.

10. What are the visualizations necessary to extract insight from the dataset? Provide a list of these steps, perform and derive the necessary insights.

In [16]:
```
import matplotlib.pyplot as plt
```

In [19]:
```
# A visualization that can be extracted from the dataset is to do a histogram
plt.hist(sales["Quantity"], sales["Product"], inplace = True)
plt.show()
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
File C:\ProgramData\anaconda3\Lib\site-packages\matplotlib\axis.py:1769, in Axis.con
vert_units(self, x)
   1768 try:
-> 1769     ret = self.converter.convert(x, self.units, self)
   1770 except Exception as e:

File C:\ProgramData\anaconda3\Lib\site-packages\matplotlib\category.py:49, in StrCat
egoryConverter.convert(value, unit, axis)
     48 if unit is None:
---> 49     raise ValueError(
     50         'Missing category information for StrCategoryConverter; '
     51         'this might be caused by unintendedly mixing categorical and '
     52         'numeric data')
     53 StrCategoryConverter._validate_unit(unit)

ValueError: Missing category information for StrCategoryConverter; this might be cau
sed by unintendedly mixing categorical and numeric data

The above exception was the direct cause of the following exception:

ConversionError                           Traceback (most recent call last)
Cell In[19], line 1
----> 1 plt.hist(sales["Quantity"], sales["Product"], inplace = True)
      2 plt.show()

File C:\ProgramData\anaconda3\Lib\site-packages\matplotlib\pyplot.py:3236, in hist
(x, bins, range, density, weights, cumulative, bottom, histtype, align, orientation,
rwidth, log, color, label, stacked, data, **kwargs)
   3211 @_copy_docstring_and_deprecators(Axes.hist)
   3212 def hist(
   3213     x: ArrayLike | Sequence[ArrayLike],
   (...)
   3234     BarContainer | Polygon | list[BarContainer | Polygon],
   3235 ]:
-> 3236     return gca().hist(
   3237         x,
   3238         bins=bins,
   3239         range=range,
   3240         density=density,
   3241         weights=weights,
   3242         cumulative=cumulative,
   3243         bottom=bottom,
   3244         histtype=histtype,
   3245         align=align,
   3246         orientation=orientation,
   3247         rwidth=rwidth,
   3248         log=log,
   3249         color=color,
   3250         label=label,
   3251         stacked=stacked,
   3252         **({"data": data} if data is not None else {}),
   3253         **kwargs,
   3254     )
```

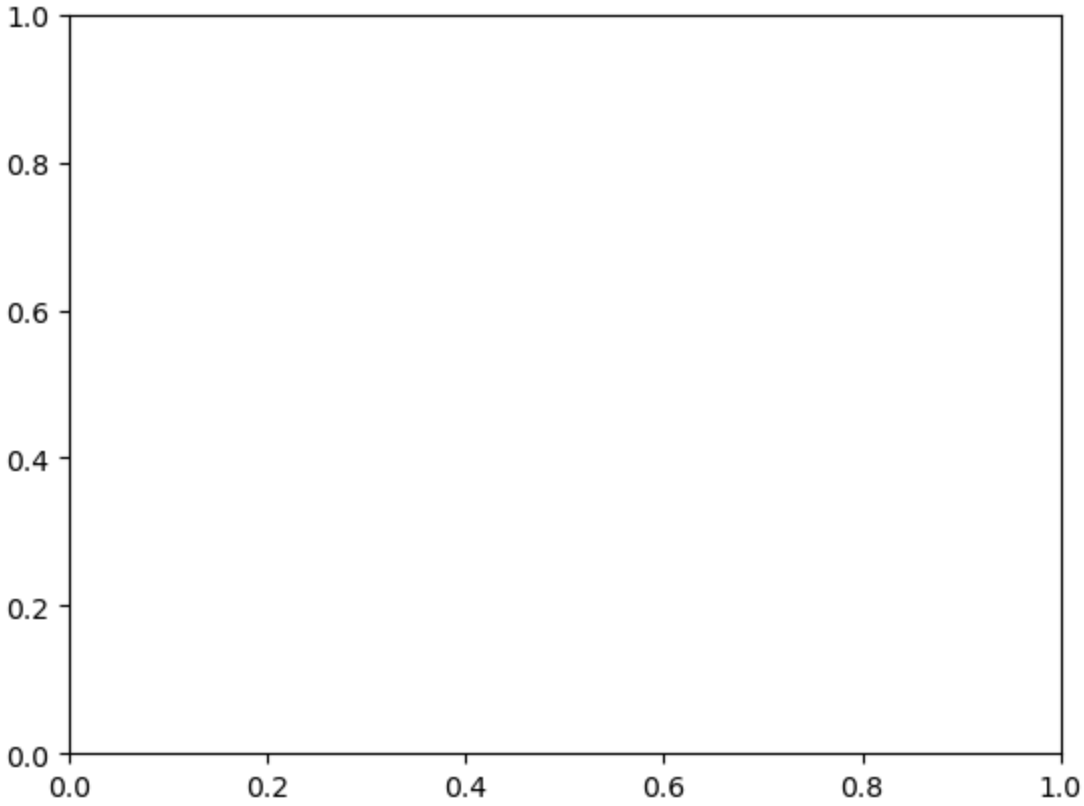
```
File C:\ProgramData\anaconda3\Lib\site-packages\matplotlib\__init__.py:1465, in _pre
process_data.<locals>.inner(ax, data, *args, **kwargs)
   1462 @functools.wraps(func)
   1463 def inner(ax, *args, data=None, **kwargs):
   1464     if data is None:
-> 1465         return func(ax, *map(sanitize_sequence, args), **kwargs)
   1467     bound = new_sig.bind(ax, *args, **kwargs)
   1468     auto_label = (bound.arguments.get(label_namer)
   1469                   or bound.kwargs.get(label_namer))

File C:\ProgramData\anaconda3\Lib\site-packages\matplotlib\axes\_axes.py:6794, in Ax
es.hist(self, x, bins, range, density, weights, cumulative, bottom, histtype, align,
orientation, rwidth, log, color, label, stacked, **kwargs)
   6791     bin_range = convert_units(bin_range)
   6793 if not cbook.is_scalar_or_string(bins):
-> 6794     bins = convert_units(bins)
   6796 # We need to do to 'weights' what was done to 'x'
   6797 if weights is not None:

File C:\ProgramData\anaconda3\Lib\site-packages\matplotlib\artist.py:279, in Artist.
convert_xunits(self, x)
    277 if ax is None or ax.xaxis is None:
    278     return x
--> 279 return ax.xaxis.convert_units(x)

File C:\ProgramData\anaconda3\Lib\site-packages\matplotlib\axis.py:1771, in Axis.con
vert_units(self, x)
   1769     ret = self.converter.convert(x, self.units, self)
   1770 except Exception as e:
-> 1771     raise munits.ConversionError('Failed to convert value(s) to axis '
   1772                                  f'units: {x!r}') from e
   1773 return ret

ConversionError: Failed to convert value(s) to axis units: 0    Widget
1    Gadget
2    Widget
3     Gizmo
4    Widget
5    Gadget
6    Widget
7     Gizmo
8    Widget
9    Gadget
Name: Product, dtype: object
```



In [ ]: