```
Python 3.11.7 | packaged by Anaconda, Inc. | (main, Dec 15 2023, 18:05:47) [MSC v.1916 64
bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 8.20.0 -- An enhanced Interactive Python.

In [1]: import numpy as np
   ...: import pandas as pd

In [2]: # Originating Pieces files - merge these two

In [3]: OriginatingPieces1 = pd.read_csv("C:/Users/andie/OneDrive/Documents/Capstone/
Mail_v2/Piece Output v2/Originating Pieces pt.1 v2.csv")

In [4]: OriginatingPieces2 = pd.read_csv("C:/Users/andie/OneDrive/Documents/Capstone/
Mail_v2/Piece Output v2/Originating Pieces pt.2 v2.csv")

In [5]: #------- Filter out data by ACTUAL_DLVRY_DATE that isn't between 01/08/24 to
01/21/24

In [6]: op_january = OriginatingPieces1[(OriginatingPieces1['ACTUAL_DLVRY_DATE'] >
"2024-01-08") & (OriginatingPieces1['ACTUAL_DLVRY_DATE'] < "2024-01-21")]

In [7]: op2_january = OriginatingPieces2[(OriginatingPieces2['ACTUAL_DLVRY_DATE'] >
"2024-01-08") & (OriginatingPieces2['ACTUAL_DLVRY_DATE'] < "2024-01-21")]

In [8]: # Clean the two data sets

In [9]: noNull_op1 = op_january.dropna(how='any',axis=0)

In [10]: clean_op1 = noNull_op1.drop_duplicates()

In [11]: noNull_op2 = op2_january.dropna(how='any',axis=0)

In [12]: clean_op2 = noNull_op2.drop_duplicates()

In [13]: # Add columns that determine if the mail was On Time Exactly, Early, or Late

In [14]: clean_op1['OnTimeExactly'] = np.where(clean_op1['ACTUAL_DLVRY_DATE'] ==
clean_op1['EXPECTED_DELIVERY_DATE'], True, False)

In [15]: clean_op1['Early'] = np.where(clean_op1['ACTUAL_DLVRY_DATE'] <
clean_op1['EXPECTED_DELIVERY_DATE'], True, False)

In [16]: clean_op1['Late'] = np.where(clean_op1['ACTUAL_DLVRY_DATE'] >
clean_op1['EXPECTED_DELIVERY_DATE'], True, False)

In [17]: clean_op2['OnTimeExactly'] = np.where(clean_op2['ACTUAL_DLVRY_DATE'] ==
clean_op2['EXPECTED_DELIVERY_DATE'], True, False)

In [18]: clean_op2['Early'] = np.where(clean_op2['ACTUAL_DLVRY_DATE'] <
clean_op2['EXPECTED_DELIVERY_DATE'], True, False)

In [19]: clean_op2['Late'] = np.where(clean_op2['ACTUAL_DLVRY_DATE'] >
clean_op2['EXPECTED_DELIVERY_DATE'], True, False)

In [20]: # Combine the datasets with only data between Jan 8th and Jan 21st of 2024

In [21]: originatingPieces_stormPeriod = pd.concat([clean_op1, clean_op2])

In [22]: # Compare the delivery status of the mail (Late vs. Ontime / Early)

In [23]: originating_late = originatingPieces_stormPeriod['Late'].values.sum()
```

```
In [24]: originating_early = originatingPieces_stormPeriod['Early'].values.sum()

In [25]: originating_ontime = originatingPieces_stormPeriod['OnTimeExactly'].values.sum()

In [26]: print("Late mail: "+ str(originating_late) + ", Early Mail: "+
str(originating_early)+", On Time Mail: "+ str(originating_ontime))
Late mail: 2755583, Early Mail: 17054971, On Time Mail: 8634603

In [27]: # ratio of late vs rest

In [28]: originating_late_ratio = originating_late / (originating_early +
originating_ontime)

In [29]: print("Ratio of Late Mail vs. Early or On Time mail: " + str("%f" %
originating_late_ratio))
Ratio of Late Mail vs. Early or On Time mail: 0.107265

In [30]: # distribution of mail shape by lateness

In [31]: originatingPieces_stormPeriod['MAIL_SHAPE'].value_counts()
Out[31]:
MAIL_SHAPE
Letter    25793548
Flat       2376958
Card        274651
Name: count, dtype: int64

In [32]: op_shape = originatingPieces_stormPeriod.groupby(by=["MAIL_SHAPE", "Late"]).size()
# True = Late, False = either Early or On Time Exactly

In [33]: print(op_shape)
MAIL_SHAPE  Late
Card        False      172241
            True       102410
Flat        False     2235735
            True       141223
Letter      False    23281598
            True      2511950
dtype: int64

In [34]: # Mail Class

In [35]: originatingPieces_stormPeriod['MAIL_CLASS'].value_counts()
Out[35]:
MAIL_CLASS
USPS Marketing Mail         17134116
First Class Presort          6163077
Single Piece First Class     4599518
Periodicals                   548446
Name: count, dtype: int64

In [36]: op_class = originatingPieces_stormPeriod.groupby(by=["MAIL_CLASS", "Late"]).size()
# True = Late, False = either Early or On Time Exactly

In [37]: print(op_class)
MAIL_CLASS                Late
First Class Presort       False     5190341
                          True       972736
Periodicals               False      521124
                          True        27322
Single Piece First Class  False     3788133
                          True       811385
```

```
USPS Marketing Mail       False    16189976
                          True       944140
dtype: int64

In [38]: #ORIGIN_FACILITY

In [39]: originatingPieces_stormPeriod['ORIGIN_FACILITY'].value_counts()
Out[39]:
ORIGIN_FACILITY
NASHVILLE - 1441275           13046409
MEMPHIS NDC - 1372672          7163871
MEMPHIS - 1441274              6071953
MUSIC CITY ANNEX - 1532174     2162924
Name: count, dtype: int64

In [40]: op_originFacility = originatingPieces_stormPeriod.groupby(by=["ORIGIN_FACILITY",
"Late"]).size() # True = Late, False = either Early or On Time Exactly

In [41]: print(op_originFacility)
ORIGIN_FACILITY             Late
MEMPHIS - 1441274           False     4807851
                            True      1264102
MEMPHIS NDC - 1372672       False     6520111
                            True       643760
MUSIC CITY ANNEX - 1532174  False     2053387
                            True       109537
NASHVILLE - 1441275         False    12308225
                            True       738184
dtype: int64

In [42]: # Convert delivery date columns to the 'date' data type

In [43]: originatingPieces_stormPeriod['ACTUAL_DLVRY_DATE'] =
pd.to_datetime(originatingPieces_stormPeriod['ACTUAL_DLVRY_DATE'])

In [44]: print(originatingPieces_stormPeriod['ACTUAL_DLVRY_DATE'].head())
0   2024-01-16
1   2024-01-13
2   2024-01-16
3   2024-01-16
4   2024-01-16
Name: ACTUAL_DLVRY_DATE, dtype: datetime64[ns]

In [45]: originatingPieces_stormPeriod['EXPECTED_DELIVERY_DATE'] =
pd.to_datetime(originatingPieces_stormPeriod['EXPECTED_DELIVERY_DATE'])

In [46]: # Difference between EXPECTED and ACTUAL delivery dates - Positive values indicate
LATE deliveries

In [47]: originatingPieces_stormPeriod['Difference'] =
(originatingPieces_stormPeriod['ACTUAL_DLVRY_DATE'] -
originatingPieces_stormPeriod['EXPECTED_DELIVERY_DATE']).dt.days

In [48]: # Average days late a piece of mail arrives

In [49]: late_deliveries =
originatingPieces_stormPeriod.loc[originatingPieces_stormPeriod.Late]

In [50]: latemean = late_deliveries['Difference'].mean()

In [51]: print("The mean for mail delivered late is: " + str("%f" % latemean) + " days after
expected delivery date")
The mean for mail delivered late is: 1.955629 days after expected delivery date
```

```
In [52]: # Differences and their Count by Mail Class

In [53]: late_by_class = late_deliveries.groupby(by=["MAIL_CLASS", "Difference"]).size() #
True = Late, False = either Early or On Time Exactly

In [54]: print(late_by_class)
MAIL_CLASS                Difference
First Class Presort       1             457482
                          2             105264
                          3             220840
                          4             140535
                          5              39176
                          6               5482
                          7               2202
                          8                965
                          9                663
                          10               113
                          11                14
Periodicals               1              18051
                          2               5506
                          3               1653
                          4               1110
                          5                440
                          6                238
                          7                140
                          8                118
                          9                 60
                          10                 6
Single Piece First Class  1             366897
                          2             127668
                          3             170358
                          4              86932
                          5              31786
                          6              14106
                          7               7623
                          8               3876
                          9               1619
                          10               520
USPS Marketing Mail       1             694905
                          2             156839
                          3              52583
                          4              21396
                          5               7277
                          6               5131
                          7               4004
                          8               1443
                          9                562
dtype: int64

In [55]: # Differences and their Count by Mail Shape

In [56]: late_by_shape = late_deliveries.groupby(by=["MAIL_SHAPE", "Difference"]).size() #
True = Late, False = either Early or On Time Exactly

In [57]: print(late_by_shape)
MAIL_SHAPE  Difference
Card        1               29796
            2               13202
            3               26948
            4               27079
            5                3062
            6                1110
```

```
            7                1033
            8                 155
            9                  23
            10                  2
Flat        1               76032
            2               28747
            3               19968
            4                9575
            5                3503
            6                1333
            7                1412
            8                 493
            9                 148
            10                 12
Letter      1             1431507
            2              353328
            3              398518
            4              213319
            5               72114
            6               22514
            7               11524
            8                5754
            9                2733
            10                625
            11                 14
dtype: int64
```

In [**58**]: