# Content

## Task 5.1

*Python is used for this task with Pandas and NumPy libraries.*

*The columns [Flags] and [Flag Text] are already obtained in Task 1, so they are used directly for this task.*

First read the csv file, and to facilitate further operations, the [Date] column is converted to a datetime type. Create a new column called [Hour], which store the specific hour number corresponding to column [Date]. Select the data that meets the criteria (Tuesday between 7:00 and 19:00) and store it in the tue_data variable.

```
data = pd.read_csv("rawpvr_2018-02-01_28d_1083 TueFri.csv")
data["Date"] = pd.to_datetime(data["Date"], format="%Y-%m-%d %H:%M:%S.%f")

data["Hour"] = data["Date"].dt.hour

tue_data = data[(data["Flags"]==2) & (data["Hour"]>=7) & (data["Hour"]<19)]
```

***To compute the column completeness of [Gap (s)],*** choose [Gap (s)] column from tue_data, get the values of each cell and make them as an array. Then get the length of this array, which is the number of cells. By applying np.isnan().sum() to the array, the length of cells that is empty is obtained.

```
gaps_data_array = tue_data["Gap (s)"].values
print("Gap (s) values array", gaps_data_array)
gaps_data_len = len(gaps_data_array)
gaps_data_nan_len = np.isnan(gaps_data_array).sum()
print("Number of cells ", gaps_data_len)
print("Number of empty cells", gaps_data_nan_len)
```

✓ 0.3s

```
Gap (s) values array [  nan   nan 1.992 ... 5.182 7.307 1.133]
Number of cells  201125
Number of empty cells 3949
```

Finally, use number of cells minus empty cells to get the number of non-empty cells, then divided by number of cells to obtain the column completeness of [Gap (s)].

```
gaps_column_completeness = ((gaps_data_len - gaps_data_nan_len)*100) / gaps_data_len
print("The completeness of the \'Gap (s)\' column ", gaps_column_completeness)
```

✓ 0.3s

```
The completeness of the 'Gap (s)' column  98.03654443753885
```

***Result***: Completeness of [Gap (s)]: 98.03654443753885

***To compute the column completeness of [Headway (s)],*** another function count() is

used and applied to tue_data, it return the number of cells without counting nan cells.

```
headways_data_no_nan_len = tue_data.count()["Headway (s)"]
headways_data_len = len(tue_data)
print("Number of non-empty cells", headways_data_no_nan_len)
print("Number of cells", headways_data_len)
```

✓ 0.6s

```
Number of non-empty cells 199044
Number of cells 201125
```

Finally using number of non-empty cells divided by number of cells to obtain the column completeness of [Headway (s)].

```
headways_column_completeness = (headways_data_no_nan_len*100) / headways_data_len
print("The level of completeness of the \'Headway (s) \' column: ", headways_column_completeness)
```

✓ 0.3s

```
The level of completeness of the 'Headway (s) ' column:  98.96532007458049
```

***Result***: Completeness of [Headway (s)]: 98.96532007458049

***Task 5.1 Result Summary (2 Decimal):***

| | Gap (s) | Headway(s) |
|---|---|---|
| Column Completeness | 98.04 | 98.97 |

***Task 5.1 Interpretation:***

Completeness is one of the characteristics of data quality. both [Gap (s)] and [Headway (s)] have a column data completeness above 98, indicating that they have very high data completeness and that most of the data is recorded. However, this also means that there is some data that is still empty and data cleaning can be performed to improve data quality and completeness.

# Task 5.2

*Python is used for this task with Pandas and NumPy libraries.*

*The columns [Flags] and [Flag Text] are already obtained in Task 1, so they are used directly for this task.*

First read the csv file, and to facilitate further operations, the [Date] column is converted to a datetime type. Create a new column called [Hour], which store the specific hour number corresponding to column [Date]. Select the data that meets the criteria (Tuesday between 7:00 and 19:00, NB_MID lane) and store it in the tue_data variable.

```
data = pd.read_csv("rawpvr_2018-02-01_28d_1083 TueFri.csv")
data["Date"] = pd.to_datetime(data["Date"], format="%Y-%m-%d %H:%M:%S.%f")
data["Hour"] = data["Date"].dt.hour
tue_data = data[(data["Direction Name"]=="North") & (data["Lane Name"]=="NB_MID") & (data["Hour"]>=7) & (data["Hour"]<19) & (data["Flags"]==2)]
```

***To fill the missing values of column [Gap (s)]***, it is necessary to sort the data of each

hour in [Gap (s)] to get the median values of each hour. First approach is by doing this manually. Define a for loop from 7 to 19 that represent each hour to get 12 arrays of [Gap (s)] of each hour, and sort them. If array length is odd then the median value is the value at the position of half of the array, otherwise the median value is the two values at the position of half of the array values added then divided by 2.

```
gaps_median_array = []
for i in range(7,19):
    gaps_data_sorted_array = tue_data[tue_data["Hour"]==i].dropna(axis=0, how="any", subset=["Gap (s)"]).sort_values("Gap (s)")["Gap (s)"].values
    if len(gaps_data_sorted_array) % 2 == 1:
        temp_median = gaps_data_sorted_array[int(len(gaps_data_sorted_array)/2)]
    elif len(gaps_data_sorted_array) % 2 == 0:
        temp_median = (gaps_data_sorted_array[int(len(gaps_data_sorted_array)/2-1)] + gaps_data_sorted_array[int(len(gaps_data_sorted_array)/2)])/2

    print("Median value of Gap (s) between %d:00 and %d:00: " % (i, i+1), temp_median)
    gaps_median_array.append(temp_median)
```

✓ 0.1s

```
Median value of Gap (s) between 7:00 and 8:00:  1.834
Median value of Gap (s) between 8:00 and 9:00:  2.1020000000000003
Median value of Gap (s) between 9:00 and 10:00:  2.08
Median value of Gap (s) between 10:00 and 11:00:  2.5835
Median value of Gap (s) between 11:00 and 12:00:  2.7785
Median value of Gap (s) between 12:00 and 13:00:  2.795
Median value of Gap (s) between 13:00 and 14:00:  2.7560000000000002
Median value of Gap (s) between 14:00 and 15:00:  2.8049999999999997
Median value of Gap (s) between 15:00 and 16:00:  2.577
Median value of Gap (s) between 16:00 and 17:00:  2.3225
Median value of Gap (s) between 17:00 and 18:00:  2.187
Median value of Gap (s) between 18:00 and 19:00:  2.228
```

Finally, get the row index of the missing values, then these indexes are used to locate and fill in with the median value corresponding to specific hour.

```
for i in range(7,19):
    temp_fill_in_index = list(tue_data[(np.isnan(tue_data["Gap (s)"])) & (tue_data["Hour"]==i)].index)
    data.loc[temp_fill_in_index,"Gap (s)"] = gaps_median_array[i-7]
```

***Result***:

The table of [Gap (s)] median values of each hour:

| Hour | Gap (s) | Hour | Gap (s) |
|---|---|---|---|
| 7 | 1.834 | 13 | 2.7560000000000002 |
| 8 | 2.1020000000000003 | 14 | 2.8049999999999997 |
| 9 | 2.08 | 15 | 2.577 |
| 10 | 2.5835 | 16 | 2.3225 |
| 11 | 2.7785 | 17 | 2.187 |
| 12 | 2.795 | 18 | 2.228 |

Here are some example screenshots ([Hour]=7,8,12,13,17,18) of first 5 rows before and after filling in.

```
tue_data.loc[tue_data[(np.isnan(tue_data["Gap (s)"]) & (tue_data["Hour"]==7))].index].head(5)
```

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 68517 | 2018-02-06 07:00:01.160 | 2 | NB_MID | 1 | North | 27.962 | 1.720 | NaN | 2 | Tuesday | 7 |
| 68547 | 2018-02-06 07:00:15.150 | 2 | NB_MID | 1 | North | 27.340 | 1.841 | NaN | 2 | Tuesday | 7 |
| 68888 | 2018-02-06 07:05:29.000 | 2 | NB_MID | 1 | North | 34.798 | NaN | NaN | 2 | Tuesday | 7 |
| 69204 | 2018-02-06 07:10:20.020 | 2 | NB_MID | 1 | North | 35.417 | NaN | NaN | 2 | Tuesday | 7 |
| 69653 | 2018-02-06 07:15:16.160 | 2 | NB_MID | 1 | North | 51.575 | 3.947 | NaN | 2 | Tuesday | 7 |

```
data.loc[tue_data[(np.isnan(tue_data["Gap (s)"]) & (tue_data["Hour"]==7))].index].head(5)
```

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 68517 | 2018-02-06 07:00:01.160 | 2 | NB_MID | 1 | North | 27.962 | 1.720 | 1.834 | 2 | Tuesday | 7 |
| 68547 | 2018-02-06 07:00:15.150 | 2 | NB_MID | 1 | North | 27.340 | 1.841 | 1.834 | 2 | Tuesday | 7 |
| 68888 | 2018-02-06 07:05:29.000 | 2 | NB_MID | 1 | North | 34.798 | NaN | 1.834 | 2 | Tuesday | 7 |
| 69204 | 2018-02-06 07:10:20.020 | 2 | NB_MID | 1 | North | 35.417 | NaN | 1.834 | 2 | Tuesday | 7 |
| 69653 | 2018-02-06 07:15:16.160 | 2 | NB_MID | 1 | North | 51.575 | 3.947 | 1.834 | 2 | Tuesday | 7 |

```
tue_data.loc[tue_data[(np.isnan(tue_data["Gap (s)"]) & (tue_data["Hour"]==8))].index].head(5)
```

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 73854 | 2018-02-06 08:00:02.000 | 2 | NB_MID | 1 | North | 16.777 | 3.267 | NaN | 2 | Tuesday | 8 |
| 73875 | 2018-02-06 08:00:16.000 | 2 | NB_MID | 1 | North | 16.155 | 3.392 | NaN | 2 | Tuesday | 8 |
| 74353 | 2018-02-06 08:05:40.000 | 2 | NB_MID | 1 | North | 17.399 | NaN | NaN | 2 | Tuesday | 8 |
| 74789 | 2018-02-06 08:10:15.150 | 2 | NB_MID | 1 | North | 15.533 | 2.664 | NaN | 2 | Tuesday | 8 |
| 75106 | 2018-02-06 08:13:55.040 | 2 | NB_MID | 1 | North | 6.214 | 0.900 | NaN | 2 | Tuesday | 8 |

+ Code + Markdown

```
data.loc[tue_data[(np.isnan(tue_data["Gap (s)"]) & (tue_data["Hour"]==8))].index].head(5)
```

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 73854 | 2018-02-06 08:00:02.000 | 2 | NB_MID | 1 | North | 16.777 | 3.267 | 2.102 | 2 | Tuesday | 8 |
| 73875 | 2018-02-06 08:00:16.000 | 2 | NB_MID | 1 | North | 16.155 | 3.392 | 2.102 | 2 | Tuesday | 8 |
| 74353 | 2018-02-06 08:05:40.000 | 2 | NB_MID | 1 | North | 17.399 | NaN | 2.102 | 2 | Tuesday | 8 |
| 74789 | 2018-02-06 08:10:15.150 | 2 | NB_MID | 1 | North | 15.533 | 2.664 | 2.102 | 2 | Tuesday | 8 |
| 75106 | 2018-02-06 08:13:55.040 | 2 | NB_MID | 1 | North | 6.214 | 0.900 | 2.102 | 2 | Tuesday | 8 |

```
tue_data.loc[tue_data[(np.isnan(tue_data["Gap (s)"]) & (tue_data["Hour"]==12))].index].head(5)
```
✓ 0.5s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 89354 | 2018-02-06 12:00:00.110 | 2 | NB_MID | 1 | North | 32.932 | 4.075 | NaN | 2 | Tuesday | 12 |
| 89372 | 2018-02-06 12:00:19.090 | 2 | NB_MID | 1 | North | 38.525 | NaN | NaN | 2 | Tuesday | 12 |
| 89623 | 2018-02-06 12:05:23.050 | 2 | NB_MID | 1 | North | 41.010 | NaN | NaN | 2 | Tuesday | 12 |
| 89908 | 2018-02-06 12:10:15.110 | 2 | NB_MID | 1 | North | 35.417 | NaN | NaN | 2 | Tuesday | 12 |
| 90216 | 2018-02-06 12:15:16.020 | 2 | NB_MID | 1 | North | 36.661 | 2.502 | NaN | 2 | Tuesday | 12 |

```
data.loc[tue_data[(np.isnan(tue_data["Gap (s)"]) & (tue_data["Hour"]==12))].index].head(5)
```
✓ 0.5s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 89354 | 2018-02-06 12:00:00.110 | 2 | NB_MID | 1 | North | 32.932 | 4.075 | 2.795 | 2 | Tuesday | 12 |
| 89372 | 2018-02-06 12:00:19.090 | 2 | NB_MID | 1 | North | 38.525 | NaN | 2.795 | 2 | Tuesday | 12 |
| 89623 | 2018-02-06 12:05:23.050 | 2 | NB_MID | 1 | North | 41.010 | NaN | 2.795 | 2 | Tuesday | 12 |
| 89908 | 2018-02-06 12:10:15.110 | 2 | NB_MID | 1 | North | 35.417 | NaN | 2.795 | 2 | Tuesday | 12 |
| 90216 | 2018-02-06 12:15:16.020 | 2 | NB_MID | 1 | North | 36.661 | 2.502 | 2.795 | 2 | Tuesday | 12 |

```
tue_data.loc[tue_data[(np.isnan(tue_data["Gap (s)"]) & (tue_data["Hour"]==13))].index].head(5)
```
✓ 0.4s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 92784 | 2018-02-06 13:00:01.170 | 2 | NB_MID | 1 | North | 32.310 | 1.627 | NaN | 2 | Tuesday | 13 |
| 92799 | 2018-02-06 13:00:16.160 | 2 | NB_MID | 1 | North | 29.205 | 2.336 | NaN | 2 | Tuesday | 13 |
| 93104 | 2018-02-06 13:05:16.030 | 2 | NB_MID | 1 | North | 32.932 | 2.785 | NaN | 2 | Tuesday | 13 |
| 93413 | 2018-02-06 13:10:16.000 | 2 | NB_MID | 1 | North | 27.962 | 1.800 | NaN | 2 | Tuesday | 13 |
| 93720 | 2018-02-06 13:15:19.010 | 2 | NB_MID | 1 | North | 30.447 | 5.437 | NaN | 2 | Tuesday | 13 |

```
data.loc[tue_data[(np.isnan(tue_data["Gap (s)"]) & (tue_data["Hour"]==13))].index].head(5)
```
✓ 0.5s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 92784 | 2018-02-06 13:00:01.170 | 2 | NB_MID | 1 | North | 32.310 | 1.627 | 2.756 | 2 | Tuesday | 13 |
| 92799 | 2018-02-06 13:00:16.160 | 2 | NB_MID | 1 | North | 29.205 | 2.336 | 2.756 | 2 | Tuesday | 13 |
| 93104 | 2018-02-06 13:05:16.030 | 2 | NB_MID | 1 | North | 32.932 | 2.785 | 2.756 | 2 | Tuesday | 13 |
| 93413 | 2018-02-06 13:10:16.000 | 2 | NB_MID | 1 | North | 27.962 | 1.800 | 2.756 | 2 | Tuesday | 13 |
| 93720 | 2018-02-06 13:15:19.010 | 2 | NB_MID | 1 | North | 30.447 | 5.437 | 2.756 | 2 | Tuesday | 13 |

```
tue_data.loc[tue_data[(np.isnan(tue_data["Gap (s)"]) & (tue_data["Hour"]==17))].index].head(5)
```
✓ 0.5s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 110218 | 2018-02-06 17:00:00.010 | 2 | NB_MID | 1 | North | 26.098 | 1.500 | NaN | 2 | Tuesday | 17 |
| 110251 | 2018-02-06 17:00:17.020 | 2 | NB_MID | 1 | North | 22.991 | 4.962 | NaN | 2 | Tuesday | 17 |
| 110448 | 2018-02-06 17:02:21.000 | 2 | NB_MID | 1 | North | 7.456 | 2.400 | NaN | 2 | Tuesday | 17 |
| 110595 | 2018-02-06 17:03:57.080 | 2 | NB_MID | 1 | North | 21.126 | 1.853 | NaN | 2 | Tuesday | 17 |
| 110712 | 2018-02-06 17:05:17.020 | 2 | NB_MID | 1 | North | 28.584 | 1.996 | NaN | 2 | Tuesday | 17 |

```
data.loc[tue_data[(np.isnan(tue_data["Gap (s)"]) & (tue_data["Hour"]==17))].index].head(5)
```
✓ 0.4s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 110218 | 2018-02-06 17:00:00.010 | 2 | NB_MID | 1 | North | 26.098 | 1.500 | 2.187 | 2 | Tuesday | 17 |
| 110251 | 2018-02-06 17:00:17.020 | 2 | NB_MID | 1 | North | 22.991 | 4.962 | 2.187 | 2 | Tuesday | 17 |
| 110448 | 2018-02-06 17:02:21.000 | 2 | NB_MID | 1 | North | 7.456 | 2.400 | 2.187 | 2 | Tuesday | 17 |
| 110595 | 2018-02-06 17:03:57.080 | 2 | NB_MID | 1 | North | 21.126 | 1.853 | 2.187 | 2 | Tuesday | 17 |
| 110712 | 2018-02-06 17:05:17.020 | 2 | NB_MID | 1 | North | 28.584 | 1.996 | 2.187 | 2 | Tuesday | 17 |

```
tue_data.loc[tue_data[(np.isnan(tue_data["Gap (s)"]) & (tue_data["Hour"]==18))].index].head(5)
```
✓ 0.3s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 115619 | 2018-02-06 18:00:01.190 | 2 | NB_MID | 1 | North | 41.010 | NaN | NaN | 2 | Tuesday | 18 |
| 115652 | 2018-02-06 18:00:16.020 | 2 | NB_MID | 1 | North | 29.205 | 1.800 | NaN | 2 | Tuesday | 18 |
| 115895 | 2018-02-06 18:03:03.180 | 2 | NB_MID | 1 | North | 32.932 | 1.902 | NaN | 2 | Tuesday | 18 |
| 116459 | 2018-02-06 18:10:17.020 | 2 | NB_MID | 1 | North | 36.661 | NaN | NaN | 2 | Tuesday | 18 |
| 116886 | 2018-02-06 18:15:15.140 | 2 | NB_MID | 1 | North | 26.718 | 2.302 | NaN | 2 | Tuesday | 18 |

```
data.loc[tue_data[(np.isnan(tue_data["Gap (s)"]) & (tue_data["Hour"]==18))].index].head(5)
```
✓ 0.4s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 115619 | 2018-02-06 18:00:01.190 | 2 | NB_MID | 1 | North | 41.010 | NaN | 2.228 | 2 | Tuesday | 18 |
| 115652 | 2018-02-06 18:00:16.020 | 2 | NB_MID | 1 | North | 29.205 | 1.800 | 2.228 | 2 | Tuesday | 18 |
| 115895 | 2018-02-06 18:03:03.180 | 2 | NB_MID | 1 | North | 32.932 | 1.902 | 2.228 | 2 | Tuesday | 18 |
| 116459 | 2018-02-06 18:10:17.020 | 2 | NB_MID | 1 | North | 36.661 | NaN | 2.228 | 2 | Tuesday | 18 |
| 116886 | 2018-02-06 18:15:15.140 | 2 | NB_MID | 1 | North | 26.718 | 2.302 | 2.228 | 2 | Tuesday | 18 |

***To fill the missing values of column [Headway (s)]***, another approach is first group by

[Hours], then simply applying the median() function on column [Headway (s)]

```
headways_median = tue_data.groupby(["Hour"])["Headway (s)"].median()
headways_median
```

```
Hour
7     2.7220
8     3.1910
9     2.7200
10    3.0000
11    3.2100
12    3.1600
13    3.1020
14    3.1330
15    2.9200
16    2.8530
17    2.9065
18    2.7900
Name: Headway (s), dtype: float64
```

Finally, use the same way as described above to fill the median values into missing value cells.

```
headways_median_array = list(headways_median)

for i in range(7,19):
    temp_fill_in_index = list(tue_data[(np.isnan(tue_data["Headway (s)"])) & (tue_data["Hour"]==i)].index)
    data.loc[temp_fill_in_index,"Headway (s)"] = headways_median_array[i-7]
```

***Result***:

The table of [Headway (s)] median values of each hour:

| Hour | Headway (s) | Hour | Headway (s) |
|---|---|---|---|
| 7 | 2.7220 | 13 | 3.1020 |
| 8 | 3.1910 | 14 | 3.1330 |
| 9 | 2.7200 | 15 | 2.9200 |
| 10 | 3.0000 | 16 | 2.8530 |
| 11 | 3.2100 | 17 | 2.9065 |
| 12 | 3.1600 | 18 | 2.7900 |

Here are some example screenshots ([Hour]=7,8,12,13,17,18) of first 5 rows before and after filling in.

```
tue_data.loc[tue_data[(np.isnan(tue_data["Headway (s)"])) & (tue_data["Hour"]==7)].index].head(5)
```

✓ 0.5s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 68888 | 2018-02-06 07:05:29.000 | 2 | NB_MID | 1 | North | 34.798 | NaN | NaN | 2 | Tuesday | 7 |
| 69204 | 2018-02-06 07:10:20.020 | 2 | NB_MID | 1 | North | 35.417 | NaN | NaN | 2 | Tuesday | 7 |
| 72419 | 2018-02-06 07:45:27.040 | 2 | NB_MID | 1 | North | 7.456 | NaN | NaN | 2 | Tuesday | 7 |
| 72881 | 2018-02-06 07:50:17.000 | 2 | NB_MID | 1 | North | 25.476 | NaN | NaN | 2 | Tuesday | 7 |
| 195243 | 2018-02-13 07:00:23.040 | 2 | NB_MID | 1 | North | 39.768 | NaN | NaN | 2 | Tuesday | 7 |

```
data.loc[tue_data[(np.isnan(tue_data["Headway (s)"])) & (tue_data["Hour"]==7)].index].head(5)
```

✓ 0.7s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 68888 | 2018-02-06 07:05:29.000 | 2 | NB_MID | 1 | North | 34.798 | 2.722 | 1.834 | 2 | Tuesday | 7 |
| 69204 | 2018-02-06 07:10:20.020 | 2 | NB_MID | 1 | North | 35.417 | 2.722 | 1.834 | 2 | Tuesday | 7 |
| 72419 | 2018-02-06 07:45:27.040 | 2 | NB_MID | 1 | North | 7.456 | 2.722 | 1.834 | 2 | Tuesday | 7 |
| 72881 | 2018-02-06 07:50:17.000 | 2 | NB_MID | 1 | North | 25.476 | 2.722 | 1.834 | 2 | Tuesday | 7 |
| 195243 | 2018-02-13 07:00:23.040 | 2 | NB_MID | 1 | North | 39.768 | 2.722 | 1.834 | 2 | Tuesday | 7 |

```
tue_data.loc[tue_data[(np.isnan(tue_data["Headway (s)"])) & (tue_data["Hour"]==8)].index].head(5)
```

✓ 0.5s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 74353 | 2018-02-06 08:05:40.000 | 2 | NB_MID | 1 | North | 17.399 | NaN | NaN | 2 | Tuesday | 8 |
| 200149 | 2018-02-13 08:00:02.050 | 2 | NB_MID | 1 | North | 6.214 | NaN | NaN | 2 | Tuesday | 8 |
| 204738 | 2018-02-13 08:55:20.020 | 2 | NB_MID | 1 | North | 33.554 | NaN | NaN | 2 | Tuesday | 8 |

```
data.loc[tue_data[(np.isnan(tue_data["Headway (s)"])) & (tue_data["Hour"]==8)].index].head(5)
```

✓ 0.4s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 74353 | 2018-02-06 08:05:40.000 | 2 | NB_MID | 1 | North | 17.399 | 3.191 | 2.102 | 2 | Tuesday | 8 |
| 200149 | 2018-02-13 08:00:02.050 | 2 | NB_MID | 1 | North | 6.214 | 3.191 | 2.102 | 2 | Tuesday | 8 |
| 204738 | 2018-02-13 08:55:20.020 | 2 | NB_MID | 1 | North | 33.554 | 3.191 | 2.102 | 2 | Tuesday | 8 |

```
tue_data.loc[tue_data[(np.isnan(tue_data["Headway (s)"])) & (tue_data["Hour"]==12)].index].head(5)
```

✓ 0.5s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 89372 | 2018-02-06 12:00:19.090 | 2 | NB_MID | 1 | North | 38.525 | NaN | NaN | 2 | Tuesday | 12 |
| 89623 | 2018-02-06 12:05:23.050 | 2 | NB_MID | 1 | North | 41.010 | NaN | NaN | 2 | Tuesday | 12 |
| 89908 | 2018-02-06 12:10:15.110 | 2 | NB_MID | 1 | North | 35.417 | NaN | NaN | 2 | Tuesday | 12 |
| 90503 | 2018-02-06 12:20:19.030 | 2 | NB_MID | 1 | North | 34.798 | NaN | NaN | 2 | Tuesday | 12 |
| 90792 | 2018-02-06 12:25:34.090 | 2 | NB_MID | 1 | North | 42.253 | NaN | NaN | 2 | Tuesday | 12 |

```
data.loc[tue_data[(np.isnan(tue_data["Headway (s)"])) & (tue_data["Hour"]==12)].index].head(5)
```

✓ 0.6s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 89372 | 2018-02-06 12:00:19.090 | 2 | NB_MID | 1 | North | 38.525 | 3.16 | 2.795 | 2 | Tuesday | 12 |
| 89623 | 2018-02-06 12:05:23.050 | 2 | NB_MID | 1 | North | 41.010 | 3.16 | 2.795 | 2 | Tuesday | 12 |
| 89908 | 2018-02-06 12:10:15.110 | 2 | NB_MID | 1 | North | 35.417 | 3.16 | 2.795 | 2 | Tuesday | 12 |
| 90503 | 2018-02-06 12:20:19.030 | 2 | NB_MID | 1 | North | 34.798 | 3.16 | 2.795 | 2 | Tuesday | 12 |
| 90792 | 2018-02-06 12:25:34.090 | 2 | NB_MID | 1 | North | 42.253 | 3.16 | 2.795 | 2 | Tuesday | 12 |

```
tue_data.loc[tue_data[(np.isnan(tue_data["Headway (s)"])) & (tue_data["Hour"]==13)].index].head(5)
```
✓ 0.4s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 94959 | 2018-02-06 13:35:37.000 | 2 | NB_MID | 1 | North | 29.205 | NaN | NaN | 2 | Tuesday | 13 |
| 95862 | 2018-02-06 13:50:39.030 | 2 | NB_MID | 1 | North | 43.495 | NaN | NaN | 2 | Tuesday | 13 |
| 96146 | 2018-02-06 13:55:17.050 | 2 | NB_MID | 1 | North | 38.525 | NaN | NaN | 2 | Tuesday | 13 |
| 219259 | 2018-02-13 13:00:17.030 | 2 | NB_MID | 1 | North | 44.739 | NaN | NaN | 2 | Tuesday | 13 |
| 219565 | 2018-02-13 13:05:26.000 | 2 | NB_MID | 1 | North | 32.310 | NaN | NaN | 2 | Tuesday | 13 |

```
data.loc[tue_data[(np.isnan(tue_data["Headway (s)"])) & (tue_data["Hour"]==13)].index].head(5)
```
✓ 0.4s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 94959 | 2018-02-06 13:35:37.000 | 2 | NB_MID | 1 | North | 29.205 | 3.102 | 2.756 | 2 | Tuesday | 13 |
| 95862 | 2018-02-06 13:50:39.030 | 2 | NB_MID | 1 | North | 43.495 | 3.102 | 2.756 | 2 | Tuesday | 13 |
| 96146 | 2018-02-06 13:55:17.050 | 2 | NB_MID | 1 | North | 38.525 | 3.102 | 2.756 | 2 | Tuesday | 13 |
| 219259 | 2018-02-13 13:00:17.030 | 2 | NB_MID | 1 | North | 44.739 | 3.102 | 2.756 | 2 | Tuesday | 13 |
| 219565 | 2018-02-13 13:05:26.000 | 2 | NB_MID | 1 | North | 32.310 | 3.102 | 2.756 | 2 | Tuesday | 13 |

```
tue_data.loc[tue_data[(np.isnan(tue_data["Headway (s)"])) & (tue_data["Hour"]==17)].index].head(5)
```
✓ 0.4s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 237558 | 2018-02-13 17:15:22.020 | 2 | NB_MID | 1 | North | 33.554 | NaN | NaN | 2 | Tuesday | 17 |
| 238104 | 2018-02-13 17:20:57.090 | 2 | NB_MID | 1 | North | 4.970 | NaN | NaN | 2 | Tuesday | 17 |
| 363568 | 2018-02-20 17:00:06.080 | 2 | NB_MID | 1 | North | 35.417 | NaN | NaN | 2 | Tuesday | 17 |
| 366348 | 2018-02-20 17:30:25.030 | 2 | NB_MID | 1 | North | 30.447 | NaN | NaN | 2 | Tuesday | 17 |
| 368759 | 2018-02-20 17:55:22.060 | 2 | NB_MID | 1 | North | 29.825 | NaN | NaN | 2 | Tuesday | 17 |

```
data.loc[tue_data[(np.isnan(tue_data["Headway (s)"])) & (tue_data["Hour"]==17)].index].head(5)
```
✓ 0.5s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 237558 | 2018-02-13 17:15:22.020 | 2 | NB_MID | 1 | North | 33.554 | 2.9065 | 2.187 | 2 | Tuesday | 17 |
| 238104 | 2018-02-13 17:20:57.090 | 2 | NB_MID | 1 | North | 4.970 | 2.9065 | 2.187 | 2 | Tuesday | 17 |
| 363568 | 2018-02-20 17:00:06.080 | 2 | NB_MID | 1 | North | 35.417 | 2.9065 | 2.187 | 2 | Tuesday | 17 |
| 366348 | 2018-02-20 17:30:25.030 | 2 | NB_MID | 1 | North | 30.447 | 2.9065 | 2.187 | 2 | Tuesday | 17 |
| 368759 | 2018-02-20 17:55:22.060 | 2 | NB_MID | 1 | North | 29.825 | 2.9065 | 2.187 | 2 | Tuesday | 17 |

```
tue_data.loc[tue_data[(np.isnan(tue_data["Headway (s)"])) & (tue_data["Hour"]==18)].index].head(5)
```
✓ 0.4s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 115619 | 2018-02-06 18:00:01.190 | 2 | NB_MID | 1 | North | 41.010 | NaN | NaN | 2 | Tuesday | 18 |
| 116459 | 2018-02-06 18:10:17.020 | 2 | NB_MID | 1 | North | 36.661 | NaN | NaN | 2 | Tuesday | 18 |
| 117279 | 2018-02-06 18:20:21.050 | 2 | NB_MID | 1 | North | 29.825 | NaN | NaN | 2 | Tuesday | 18 |
| 118095 | 2018-02-06 18:30:17.090 | 2 | NB_MID | 1 | North | 30.447 | NaN | NaN | 2 | Tuesday | 18 |
| 118822 | 2018-02-06 18:40:23.090 | 2 | NB_MID | 1 | North | 39.768 | NaN | NaN | 2 | Tuesday | 18 |

```
data.loc[tue_data[(np.isnan(tue_data["Headway (s)"])) & (tue_data["Hour"]==18)].index].head(5)
```
✓ 0.5s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text | Hour |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 115619 | 2018-02-06 18:00:01.190 | 2 | NB_MID | 1 | North | 41.010 | 2.79 | 2.228 | 2 | Tuesday | 18 |
| 116459 | 2018-02-06 18:10:17.020 | 2 | NB_MID | 1 | North | 36.661 | 2.79 | 2.228 | 2 | Tuesday | 18 |
| 117279 | 2018-02-06 18:20:21.050 | 2 | NB_MID | 1 | North | 29.825 | 2.79 | 2.228 | 2 | Tuesday | 18 |
| 118095 | 2018-02-06 18:30:17.090 | 2 | NB_MID | 1 | North | 30.447 | 2.79 | 2.228 | 2 | Tuesday | 18 |
| 118822 | 2018-02-06 18:40:23.090 | 2 | NB_MID | 1 | North | 39.768 | 2.79 | 2.228 | 2 | Tuesday | 18 |

***Task 5.2 Result Summary (4 Decimal):***

Median values table:

| Hour | Gap (s) | Hour | Gap (s) | Hour | Headway (s) | Hour | Headway (s) |
|---|---|---|---|---|---|---|---|
| 7 | 1.834 | 13 | 2.7560 | 7 | 2.7220 | 13 | 3.1020 |
| 8 | 2.1020 | 14 | 2.8050 | 8 | 3.1910 | 14 | 3.1330 |
| 9 | 2.0800 | 15 | 2.5770 | 9 | 2.7200 | 15 | 2.9200 |
| 10 | 2.5835 | 16 | 2.3225 | 10 | 3.0000 | 16 | 2.8530 |
| 11 | 2.7785 | 17 | 2.1870 | 11 | 3.2100 | 17 | 2.9065 |
| 12 | 2.7950 | 18 | 2.2280 | 12 | 3.1600 | 18 | 2.7900 |

Result example screenshot is above

***Task 5.2 Interpretation:***

In data cleaning, it is a common operation to use the median value to fill in the empty cells. Similarly, the average value can also be used. More advanced, machine learning methods can be used to predict missing values.

From these median values, it can be predicted that the busiest time period is between 7:00 and 8:00, because the values of gap and headway are very small, indicating that the distance between the front and rear cars is very short.

For the filled table, the last [Hour] column should be deleted when writing output file, because it is not existed in the original table, it is just added for the convenience of operation.

# Task 6

*Python is used for this task with Pandas and NumPy libraries.*

*The columns [Flags] and [Flag Text] are already obtained in Task 1, so they are used directly for this task.*

First read the “1083” and “1415” csv files, and to facilitate further operations, each of the [Date] column is converted to a datetime type. Create a new column called [Hour], which store the specific hour number corresponding to column [Date]. Select the data that meets the criteria (for 1083: North direction, Friday between 17:00 and 18:00; for 1415: NorthEast direction, Friday between 17:00 and 18:00) and store them in the fri_data1083 and fri_data1415 variables respectively.

```
data1083 = pd.read_csv("rawpvr_2018-02-01_28d_1083 TueFri.csv")
data1415 = pd.read_csv("rawpvr_2018-02-01_28d_1415 TueFri.csv")
```
0.9s

```
data1083["Date"] = pd.to_datetime(data1083["Date"], format="%Y-%m-%d %H:%M:%S.%f")
data1415["Date"] = pd.to_datetime(data1415["Date"], format="%Y-%m-%d %H:%M:%S.%f")

data1083["Hour"] = data1083["Date"].dt.hour
data1415["Hour"] = data1415["Date"].dt.hour

fri_data1083 = data1083[(data1083["Direction Name"]=="North") & (data1083["Flags"]==5) & (data1083["Hour"]==17)]
fri_data1415 = data1415[(data1415["Direction Name"]=="NorthEast") & (data1415["Flags"]==5) & (data1415["Hour"]==17)]
```

Then compute the average speed of each lane

```
avg_speed_fri_data1083 = fri_data1083.groupby("Lane Name").mean()["Speed (mph)"]
avg_speed_fri_data1415 = fri_data1415.groupby("Lane Name").mean()["Speed (mph)"]
print("Average speed in 1083: ", avg_speed_fri_data1083)
print("Average speed in 1415: ,", avg_speed_fri_data1415)
```
✓ 0.5s

```
Average speed in 1083:  Lane Name
NB_MID    28.781308
NB_NS     24.889867
NB_OS     30.153217
Name: Speed (mph), dtype: float64
Average speed in 1415: , Lane Name
NE_NS    24.096575
NE_OS    27.159322
Name: Speed (mph), dtype: float64
```

Finally, compute a total average speed by using these five lanes average speed. As this speed is miles per hour (27.02 mph), convert it to kilometers per hour (43.49 kph) by multiply 1.61. Using distance 4.86km divided by kph speed, the time from site 1083 to site 1415 in hour is obtained (0.11 hours). The time from site 1083 to site 1415 in minutes can obtained by time in hour multiply by 60. (6.70 minutes)

```
speed_fri_list = list(avg_speed_fri_data1083) + list(avg_speed_fri_data1415)
avg_speed_fri_mph = sum(speed_fri_list)/len(speed_fri_list)
print("Average speed (mph): ", avg_speed_fri_mph)

avg_speed_fri_kph = avg_speed_fri_mph * 1.61
print("Average speed (kph): ", avg_speed_fri_kph)

time_hours = 4.86/avg_speed_fri_kph
time_minutes = time_hours*60
print("Time from site 1083 to site 1415 in hours", time_hours)
print("Time from site 1083 to site 1415 in minutes", time_minutes)
```
✓ 0.3s

```
Average speed (mph):  27.016057646033993
Average speed (kph):  43.495852810114734
Time from site 1083 to site 1415 in hours 0.1117347904688015
Time from site 1083 to site 1415 in minutes 6.70408742812809
```

***Task 6 Result (3 Decimal)***: Journey time: 6.704 minutes

***Task 6 Interpretation:***

The mean is used to describe the central tendency of the distribution of data. The average of five lanes is used here to estimate the travel time from site 1083 to site 1415. This shows that under normal circumstances, most vehicles take about 6.75 minutes. This can give a time estimate for travelers. Also, it can judge whether this road is congested by comparing the current travel time with the average time.

# Task 7.1

*Python is used for this task with Pandas and NumPy libraries.*

*The columns [Flags] and [Flag Text] are already obtained in Task 1, so they are used directly for this task.*

**(i)** Here defined two row completeness formulas

First is:

$$Row\ Completeness\ 1 = \frac{\sum_{i=0}^{all\ rows} \frac{non\_empty\ cells\ of\ i^{th}\ row}{all\ cells\ of\ i^{th}\ row}}{Number\ of\ rows} \times 100$$

Second is:

$$Row\ Completeness\ 2 = \frac{Number\ of\ non\_empty\ rows}{Number\ of\ rows} \times 100$$

**(ii)** First read the csv file, and to facilitate further operations, the [Date] column is converted to a datetime type. Create a new column called [Hour], which store the specific hour number corresponding to column [Date]. Select the data that meets the criteria (Tuesday between 7:00 and 19:00) and store it in the tue_data variable.

After selection, drop the column [Hour] to compute the row completeness.

```
data = pd.read_csv("rawpvr_2018-02-01_28d_1083 TueFri.csv")
data["Date"] = pd.to_datetime(data["Date"])
data["Hour"] = data["Date"].dt.hour

tue_data = data[(data["Flags"]==2) & (data["Hour"]>=7) & (data["Hour"]<19)]
tue_data = tue_data.drop(["Hour"], axis=1)
tue_data.head(3)
```

✓ 0.9s

| | Date | Lane | Lane Name | Direction | Direction Name | Speed (mph) | Headway (s) | Gap (s) | Flags | Flag Text |
|---|---|---|---|---|---|---|---|---|---|---|
| 68516 | 2018-02-06 07:00:00.100 | 3 | NB_OS | 1 | North | 32.310 | 3.462 | NaN | 2 | Tuesday |
| 68517 | 2018-02-06 07:00:01.160 | 2 | NB_MID | 1 | North | 27.962 | 1.720 | NaN | 2 | Tuesday |
| 68518 | 2018-02-06 07:00:03.000 | 2 | NB_MID | 1 | North | 27.962 | 2.480 | 1.992 | 2 | Tuesday |

***Using formula 1***:

```
accuracy = 0
for i in tue_data.index:
    accuracy += (len(tue_data.loc[i]) - tue_data.loc[i].isna().sum())/len(tue_data.loc[i])

accuracy = accuracy/len(tue_data)*100
print("Row completeness accuracy: ", accuracy)
```

```
Row completeness accuracy:  99.6997389682972
```

***Using formula 2***: First check which label has missing values.

```
tue_data.isna().sum()
```
✓ 0.5s

| | |
|---|---|
| Date | 0 |
| Lane | 0 |
| Lane Name | 0 |
| Direction | 0 |
| Direction Name | 0 |
| Speed (mph) | 9 |
| Headway (s) | 2081 |
| Gap (s) | 3949 |
| Flags | 0 |
| Flag Text | 0 |

dtype: int64

Then get the number of rows which has missing values

```
nan_data_len = len(tue_data[(np.isnan(tue_data["Speed (mph)"])) | (np.isnan(tue_data["Headway (s)"])) | (np.isnan(tue_data["Gap (s)"]))].index)
nan_data_len
```
✓ 0.4s

```
3958
```

Finally, using calculate the row completeness

```
accuracy = ((len(tue_data)-nan_data_len)/len(tue_data))*100
print("Row completeness accuracy: ", accuracy)
```
✓ 0.2s

```
Row completeness accuracy:  98.03206960845246
```

***Task 7.1 (ii) Result (2 Decimal):*** By using first formula, row completeness is 99.70; By using second formula, row completeness is 98.03.

**(iii) Discussion**

Completeness is used to describe the comprehensiveness or wholeness of the data. The full score is 100, with higher scores representing fewer missing data. This is one aspect of evaluating the quality of the data.

The first formula means the average completeness of each row, and a result of 99.70 means that the average completeness of each row is 99.70.

The second formula means how many rows are complete and the result of 98.03 means

that approximately 98 out of every 100 rows are complete.

The first formula is therefore more concerned with the completeness of each row, while the second formula is more concerned with the overall row completeness. However, the data usually requires complete rows and does not care how much data is missing from each row, or more precisely, one missing data of a row versus multiple missing data of a row does not allow for further analysis. Therefore, the second formula seems to make more sense from this point of view.

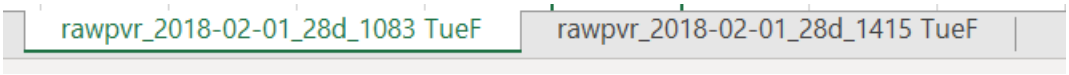
I think the row completeness of formula 2 is a better measure of file integrity than column completeness in task 5.1, because one column completeness is ill-considered, and missing values in a file could in many columns, whereas row completeness takes into account the missing values in multiple columns.

## Task 7.2

*Excel is used for this task.*

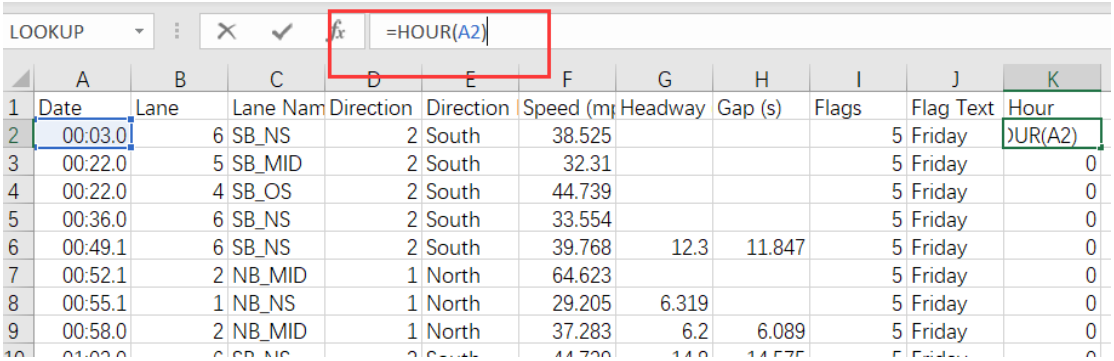
*The columns [Flags] and [Flag Text] are already obtained in Task 1, so they are used directly for this task.*

First put open two files in one single window.




Use function Hour() to get specific hour corresponding to [Date] in both 1083 and 1415 files.

1083:




| | A | B | C | D | E | F | G | H | I | J | K |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | Lane | Lane Nam | Direction | Direction | Speed (m | Headway | Gap (s) | Flags | Flag Text | Hour |
| 2 | 00:03.0 | 6 | SB_NS | 2 | South | 38.525 | | | 5 | Friday | )UR(A2) |
| 3 | 00:22.0 | 5 | SB_MID | 2 | South | 32.31 | | | 5 | Friday | 0 |
| 4 | 00:22.0 | 4 | SB_OS | 2 | South | 44.739 | | | 5 | Friday | 0 |
| 5 | 00:36.0 | 6 | SB_NS | 2 | South | 33.554 | | | 5 | Friday | 0 |
| 6 | 00:49.1 | 6 | SB_NS | 2 | South | 39.768 | 12.3 | 11.847 | 5 | Friday | 0 |
| 7 | 00:52.1 | 2 | NB_MID | 1 | North | 64.623 | | | 5 | Friday | 0 |
| 8 | 00:55.1 | 1 | NB_NS | 1 | North | 29.205 | 6.319 | | 5 | Friday | 0 |
| 9 | 00:58.0 | 2 | NB_MID | 1 | North | 37.283 | 6.2 | 6.089 | 5 | Friday | 0 |

1415:

LOOKUP | =HOUR(A2)

| | A | B | C | D | E | F | G | H | I | J | K |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | Lane | Lane Nam | Direction | Direction | Speed (mp | Headway | Gap (s) | Flags | Flag Text | Hour |
| 2 | 00:01.0 | 3 | SW | 2 | SouthWes | 26.098 | | | 5 | Friday | )UR(A2) |
| 3 | 00:03.1 | 3 | SW | 2 | SouthWes | 34.176 | 1.636 | 1.171 | 5 | Friday | 0 |
| 4 | 00:37.1 | 3 | SW | 2 | SouthWes | 24.855 | | | 5 | Friday | 0 |
| 5 | 00:40.1 | 3 | SW | 2 | SouthWes | 36.661 | 2.38 | 2.523 | 5 | Friday | 0 |
| 6 | 00:41.1 | 2 | NE_OS | 1 | NorthEast | 16.155 | | | 5 | Friday | 0 |
| 7 | 00:46.0 | 3 | SW | 2 | SouthWes | 20.506 | 6.6 | 6.307 | 5 | Friday | 0 |
| 8 | 00:53.1 | 2 | NE_OS | 1 | NorthEast | 44.739 | 4.6 | 11.346 | 5 | Friday | 0 |
| 9 | ###### | 3 | SW | 2 | SouthWes | 37.903 | 4.928 | 8.02 | 5 | Friday | 0 |
| 10 | 01:04.0 | 3 | SW | 2 | SouthWes | 39.146 | 9.3 | 8.964 | 5 | Friday | 0 |
| 11 | 01:06.0 | 2 | NE_OS | 1 | NorthEast | 22.991 | 13.5 | 13.265 | 5 | Friday | 0 |

Next, use SUMIFS() function to calculate the total speed of each lane that satisfied the condition (for 1083: Friday between 17:00 and 18:00, NB_MID lane or NB_OS lane or NB_NS lane; for 1415: Friday between 17:00 and 18:00, NE_OS lane or NE_NS lane).

N4 | =SUMIFS(F:F,E:E,"North",I:I,5,K:K,17,C:C,"NB_MID")

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | Lane | Lane Nam | Direction | Direction | Speed (mp | Headway | Gap (s) | Flags | Flag Text | Hour | | | | | |
| 2 | 00:03.0 | 6 | SB_NS | 2 | South | 38.525 | | | 5 | Friday | 0 | | | | | |
| 3 | 00:22.0 | 5 | SB_MID | 2 | South | 32.31 | | | 5 | Friday | 0 | | | NB_MID | NB_OS | NB_NS |
| 4 | 00:22.0 | 4 | SB_OS | 2 | South | 44.739 | | | 5 | Friday | 0 | | Total Speed | 91207.96 | 100108.7 | 86143.83 |
| 5 | 00:36.0 | 6 | SB_NS | 2 | South | 33.554 | | | 5 | Friday | 0 | | Average Speed | 28.78131 | 30.15322 | 24.88987 |
| 6 | 00:49.1 | 6 | SB_NS | 2 | South | 39.768 | 12.3 | 11.847 | 5 | Friday | 0 | | | | | |
| 7 | 00:52.1 | 2 | NB_MID | 1 | North | 64.623 | | | 5 | Friday | 0 | | | | | |
| 8 | 00:55.1 | 1 | NB_NS | 1 | North | 29.205 | 6.319 | | 5 | Friday | 0 | | | | | |
| 9 | 00:58.0 | 2 | NB_MID | 1 | North | 37.283 | 6.2 | 6.089 | 5 | Friday | 0 | | | | | |
| 10 | 01:03.0 | 6 | SB_NS | 2 | South | 44.739 | 14.8 | 14.575 | 5 | Friday | 0 | | | | | |

O4 | =SUMIFS(F:F,E:E,"North",I:I,5,K:K,17,C:C,"NB_OS")

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | Lane | Lane Nam | Direction | Direction | Speed (mp | Headway | Gap (s) | Flags | Flag Text | Hour | | | | | |
| 2 | 00:03.0 | 6 | SB_NS | 2 | South | 38.525 | | | 5 | Friday | 0 | | | | | |
| 3 | 00:22.0 | 5 | SB_MID | 2 | South | 32.31 | | | 5 | Friday | 0 | | | NB_MID | NB_OS | NB_NS |
| 4 | 00:22.0 | 4 | SB_OS | 2 | South | 44.739 | | | 5 | Friday | 0 | | Total Speed | 91207.96 | 100108.7 | 86143.83 |
| 5 | 00:36.0 | 6 | SB_NS | 2 | South | 33.554 | | | 5 | Friday | 0 | | Average Speed | 28.78131 | 30.15322 | 24.88987 |
| 6 | 00:49.1 | 6 | SB_NS | 2 | South | 39.768 | 12.3 | 11.847 | 5 | Friday | 0 | | | | | |
| 7 | 00:52.1 | 2 | NB_MID | 1 | North | 64.623 | | | 5 | Friday | 0 | | | | | |

P4 | =SUMIFS(F:F,E:E,"North",I:I,5,K:K,17,C:C,"NB_NS")

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | Lane | Lane Nam | Direction | Direction | Speed (mp | Headway | Gap (s) | Flags | Flag Text | Hour | | | | | |
| 2 | 00:03.0 | 6 | SB_NS | 2 | South | 38.525 | | | 5 | Friday | 0 | | | | | |
| 3 | 00:22.0 | 5 | SB_MID | 2 | South | 32.31 | | | 5 | Friday | 0 | | | NB_MID | NB_OS | NB_NS |
| 4 | 00:22.0 | 4 | SB_OS | 2 | South | 44.739 | | | 5 | Friday | 0 | | Total Speed | 91207.96 | 100108.7 | 86143.83 |
| 5 | 00:36.0 | 6 | SB_NS | 2 | South | 33.554 | | | 5 | Friday | 0 | | Average Speed | 28.78131 | 30.15322 | 24.88987 |
| 6 | 00:49.1 | 6 | SB_NS | 2 | South | 39.768 | 12.3 | 11.847 | 5 | Friday | 0 | | | | | |
| 7 | 00:52.1 | 2 | NB_MID | 1 | North | 64.623 | | | 5 | Friday | 0 | | | | | |
| 8 | 00:55.1 | 1 | NB_NS | 1 | North | 29.205 | 6.319 | | 5 | Friday | 0 | | | | | |
| 9 | 00:58.0 | 2 | NB_MID | 1 | North | 37.283 | 6.2 | 6.089 | 5 | Friday | 0 | | | | | |

N4 | =SUMIFS(F:F,E:E,"NorthEast",I:I,5,K:K,17,C:C,"NE_OS")

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | Lane | Lane Nam | Direction | Direction | Speed (mp | Headway | Gap (s) | Flags | Flag Text | Hour | | | | |
| 2 | 00:01.0 | 3 | SW | 2 | SouthWes | 26.098 | | | 5 | Friday | 0 | | | | |
| 3 | 00:03.1 | 3 | SW | 2 | SouthWes | 34.176 | 1.636 | 1.171 | 5 | Friday | 0 | | | NE_OS | NE_NS |
| 4 | 00:37.1 | 3 | SW | 2 | SouthWes | 24.855 | | | 5 | Friday | 0 | | Total Speed | 65073.74 | 64747.5 |
| 5 | 00:40.1 | 3 | SW | 2 | SouthWes | 36.661 | 2.38 | 2.523 | 5 | Friday | 0 | | Average Speed | 27.15932 | 24.09657 |
| 6 | 00:41.1 | 2 | NE_OS | 1 | NorthEast | 16.155 | | | 5 | Friday | 0 | | | | |
| 7 | 00:46.0 | 3 | SW | 2 | SouthWes | 20.506 | 6.6 | 6.307 | 5 | Friday | 0 | | | | |
| 8 | 00:53.1 | 2 | NE_OS | 1 | NorthEast | 44.739 | 4.6 | 11.346 | 5 | Friday | 0 | | | | |
| 9 | ###### | 3 | SW | 2 | SouthWes | 37.903 | 4.928 | 8.02 | 5 | Friday | 0 | | | | |

O4 =SUMIFS(F:F,E:E,"NorthEast",I:I,5,K:K,17,C:C,"NE_NS")

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | Lane | Lane Nam | Direction | Direction | Speed (mp | Headway | Gap (s) | Flags | Flag Text | Hour | | | | |
| 2 | 00:01.0 | 3 | SW | 2 | SouthWes | 26.098 | | | 5 | Friday | 0 | | | | |
| 3 | 00:03.1 | 3 | SW | 2 | SouthWes | 34.176 | 1.636 | 1.171 | 5 | Friday | 0 | | | NE_OS | NE_NS |
| 4 | 00:37.1 | 3 | SW | 2 | SouthWes | 24.855 | | | 5 | Friday | 0 | | Total Speed | 65073.74 | 64747.5 |
| 5 | 00:40.1 | 3 | SW | 2 | SouthWes | 36.661 | 2.38 | 2.523 | 5 | Friday | 0 | | Average Speed | 27.15932 | 24.09657 |
| 6 | 00:41.1 | 2 | NE_OS | 1 | NorthEast | 16.155 | | | 5 | Friday | 0 | | | | |
| 7 | 00:46.0 | 3 | SW | 2 | SouthWes | 20.506 | 6.6 | 6.307 | 5 | Friday | 0 | | | | |
| 8 | 00:53.1 | 2 | NE_OS | 1 | NorthEast | 44.739 | 4.6 | 11.346 | 5 | Friday | 0 | | | | |
| 9 | ###### | 3 | SW | 2 | SouthWes | 37.903 | 4.928 | 8.02 | 5 | Friday | 0 | | | | |
| 10 | 01:04.0 | 3 | SW | 2 | SouthWes | 39.146 | 9.3 | 8.964 | 5 | Friday | 0 | | | | |
| 11 | 01:06.0 | 2 | NE_OS | 1 | NorthEast | 22.991 | 13.5 | 13.265 | 5 | Friday | 0 | | | | |
| 12 | 01:06.0 | 3 | SW | 2 | SouthWes | 39.768 | 2.475 | 1.914 | 5 | Friday | 0 | | | | |
| 13 | 01:15.1 | 3 | SW | 2 | SouthWes | 50.331 | 8.4 | 8.124 | 5 | Friday | 0 | | | | |

Use COUNTIFS() function to calculate the number of cells that satisfied the condition, and average speed for each lane is calculated by using total speed obtained above divided by number of cells.

N5 =N4/COUNTIFS(E:E,"North",I:I,5,K:K,17,C:C,"NB_MID")

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | Lane | Lane Nam | Direction | Direction | Speed (mp | Headway | Gap (s) | Flags | Flag Text | Hour | | | | | |
| 2 | 00:03.0 | 6 | SB_NS | 2 | South | 38.525 | | | 5 | Friday | 0 | | | | | |
| 3 | 00:22.0 | 5 | SB_MID | 2 | South | 32.31 | | | 5 | Friday | 0 | | | NB_MID | NB_OS | NB_NS |
| 4 | 00:22.0 | 4 | SB_OS | 2 | South | 44.739 | | | 5 | Friday | 0 | | Total Speed | 91207.96 | 100108.7 | 86143.83 |
| 5 | 00:36.0 | 6 | SB_NS | 2 | South | 33.554 | | | 5 | Friday | 0 | | Average Speed | 28.78131 | 30.15322 | 24.88987 |
| 6 | 00:49.1 | 6 | SB_NS | 2 | South | 39.768 | 12.3 | 11.847 | 5 | Friday | 0 | | | | | |
| 7 | 00:52.1 | 2 | NB_MID | 1 | North | 64.623 | | | 5 | Friday | 0 | | | | | |
| 8 | 00:55.1 | 1 | NB_NS | 1 | North | 29.205 | 6.319 | | 5 | Friday | 0 | | | | | |
| 9 | 00:58.0 | 2 | NB_MID | 1 | North | 37.283 | 6.2 | 6.089 | 5 | Friday | 0 | | | | | |

O5 =O4/COUNTIFS(E:E,"North",I:I,5,K:K,17,C:C,"NB_OS")

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | Lane | Lane Nam | Direction | Direction | Speed (mp | Headway | Gap (s) | Flags | Flag Text | Hour | | | | | |
| 2 | 00:03.0 | 6 | SB_NS | 2 | South | 38.525 | | | 5 | Friday | 0 | | | | | |
| 3 | 00:22.0 | 5 | SB_MID | 2 | South | 32.31 | | | 5 | Friday | 0 | | | NB_MID | NB_OS | NB_NS |
| 4 | 00:22.0 | 4 | SB_OS | 2 | South | 44.739 | | | 5 | Friday | 0 | | Total Speed | 91207.96 | 100108.7 | 86143.83 |
| 5 | 00:36.0 | 6 | SB_NS | 2 | South | 33.554 | | | 5 | Friday | 0 | | Average Speed | 28.78131 | 30.15322 | 24.88987 |
| 6 | 00:49.1 | 6 | SB_NS | 2 | South | 39.768 | 12.3 | 11.847 | 5 | Friday | 0 | | | | | |
| 7 | 00:52.1 | 2 | NB_MID | 1 | North | 64.623 | | | 5 | Friday | 0 | | | | | |
| 8 | 00:55.1 | 1 | NB_NS | 1 | North | 29.205 | 6.319 | | 5 | Friday | 0 | | | | | |
| 9 | 00:58.0 | 2 | NB_MID | 1 | North | 37.283 | 6.2 | 6.089 | 5 | Friday | 0 | | | | | |

P5 =P4/COUNTIFS(E:E,"North",I:I,5,K:K,17,C:C,"NB_NS")

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | Lane | Lane Nam | Direction | Direction | Speed (mp | Headway | Gap (s) | Flags | Flag Text | Hour | | | | | |
| 2 | 00:03.0 | 6 | SB_NS | 2 | South | 38.525 | | | 5 | Friday | 0 | | | | | |
| 3 | 00:22.0 | 5 | SB_MID | 2 | South | 32.31 | | | 5 | Friday | 0 | | | NB_MID | NB_OS | NB_NS |
| 4 | 00:22.0 | 4 | SB_OS | 2 | South | 44.739 | | | 5 | Friday | 0 | | Total Speed | 91207.96 | 100108.7 | 86143.83 |
| 5 | 00:36.0 | 6 | SB_NS | 2 | South | 33.554 | | | 5 | Friday | 0 | | Average Speed | 28.78131 | 30.15322 | 24.88987 |
| 6 | 00:49.1 | 6 | SB_NS | 2 | South | 39.768 | 12.3 | 11.847 | 5 | Friday | 0 | | | | | |
| 7 | 00:52.1 | 2 | NB_MID | 1 | North | 64.623 | | | 5 | Friday | 0 | | | | | |
| 8 | 00:55.1 | 1 | NB_NS | 1 | North | 29.205 | 6.319 | | 5 | Friday | 0 | | | | | |
| 9 | 00:58.0 | 2 | NB_MID | 1 | North | 37.283 | 6.2 | 6.089 | 5 | Friday | 0 | | | | | |

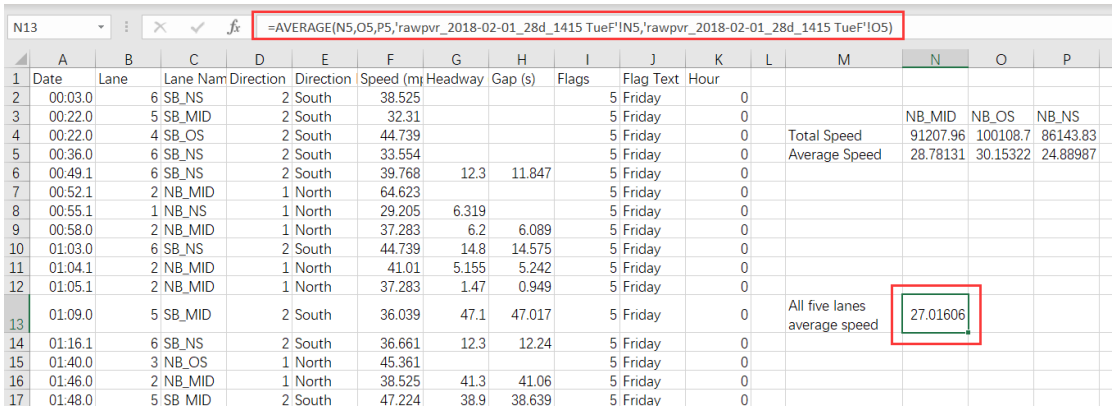
N5 =N4/COUNTIFS(E:E,"NorthEast",C:C,"NE_OS",I:I,5,K:K,17)

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | Lane | Lane Nam | Direction | Direction | Speed (mp | Headway | Gap (s) | Flags | Flag Text | Hour | | | | |
| 2 | 00:01.0 | 3 | SW | 2 | SouthWes | 26.098 | | | 5 | Friday | 0 | | | | |
| 3 | 00:03.1 | 3 | SW | 2 | SouthWes | 34.176 | 1.636 | 1.171 | 5 | Friday | 0 | | | NE_OS | NE_NS |
| 4 | 00:37.1 | 3 | SW | 2 | SouthWes | 24.855 | | | 5 | Friday | 0 | | Total Speed | 65073.74 | 64747.5 |
| 5 | 00:40.1 | 3 | SW | 2 | SouthWes | 36.661 | 2.38 | 2.523 | 5 | Friday | 0 | | Average Speed | 27.15932 | 24.09657 |
| 6 | 00:41.1 | 2 | NE_OS | 1 | NorthEast | 16.155 | | | 5 | Friday | 0 | | | | |
| 7 | 00:46.0 | 3 | SW | 2 | SouthWes | 20.506 | 6.6 | 6.307 | 5 | Friday | 0 | | | | |

O5 =O4/COUNTIFS(E:E,"NorthEast",C:C,"NE_NS",I:I,5,K:K,17)

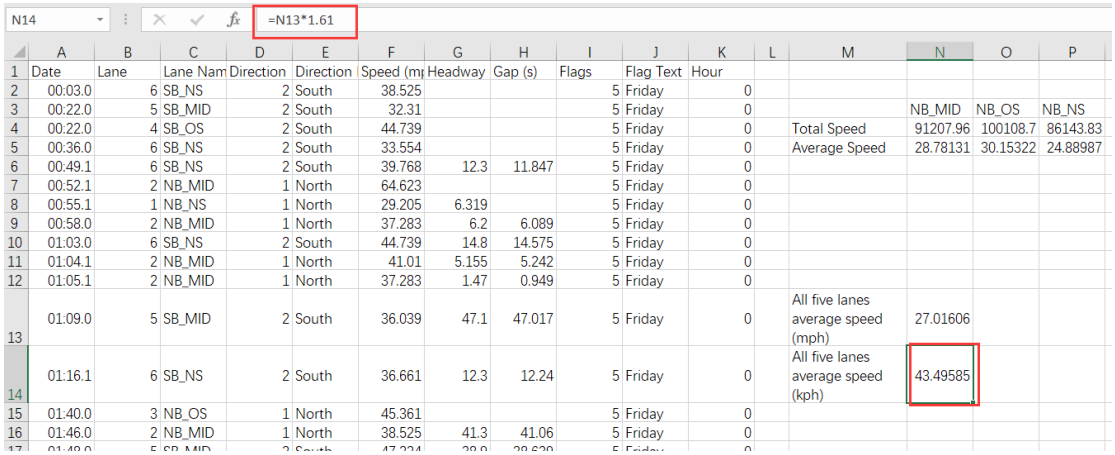
| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | Lane | Lane Nam | Direction | Direction | Speed (mp | Headway | Gap (s) | Flags | Flag Text | Hour | | | | |
| 2 | 00:01.0 | 3 | SW | 2 | SouthWes | 26.098 | | | 5 | Friday | 0 | | | | |
| 3 | 00:03.1 | 3 | SW | 2 | SouthWes | 34.176 | 1.636 | 1.171 | 5 | Friday | 0 | | | NE_OS | NE_NS |
| 4 | 00:37.1 | 3 | SW | 2 | SouthWes | 24.855 | | | 5 | Friday | 0 | | Total Speed | 65073.74 | 64747.5 |
| 5 | 00:40.1 | 3 | SW | 2 | SouthWes | 36.661 | 2.38 | 2.523 | 5 | Friday | 0 | | Average Speed | 27.15932 | 24.09657 |
| 6 | 00:41.1 | 2 | NE_OS | 1 | NorthEast | 16.155 | | | 5 | Friday | 0 | | | | |
| 7 | 00:46.0 | 3 | SW | 2 | SouthWes | 20.506 | 6.6 | 6.307 | 5 | Friday | 0 | | | | |
| 8 | 00:53.1 | 2 | NE_OS | 1 | NorthEast | 44.739 | 4.6 | 11.346 | 5 | Friday | 0 | | | | |
| 9 | ######## | 3 | SW | 2 | SouthWes | 37.903 | 4.928 | 8.02 | 5 | Friday | 0 | | | | |
| 10 | 01:04.0 | 3 | SW | 2 | SouthWes | 39.146 | 9.3 | 8.964 | 5 | Friday | 0 | | | | |
| 11 | 01:06.0 | 2 | NE_OS | 1 | NorthEast | 22.991 | 13.5 | 13.265 | 5 | Friday | 0 | | | | |

Calculate the average speed of 5 lanes.



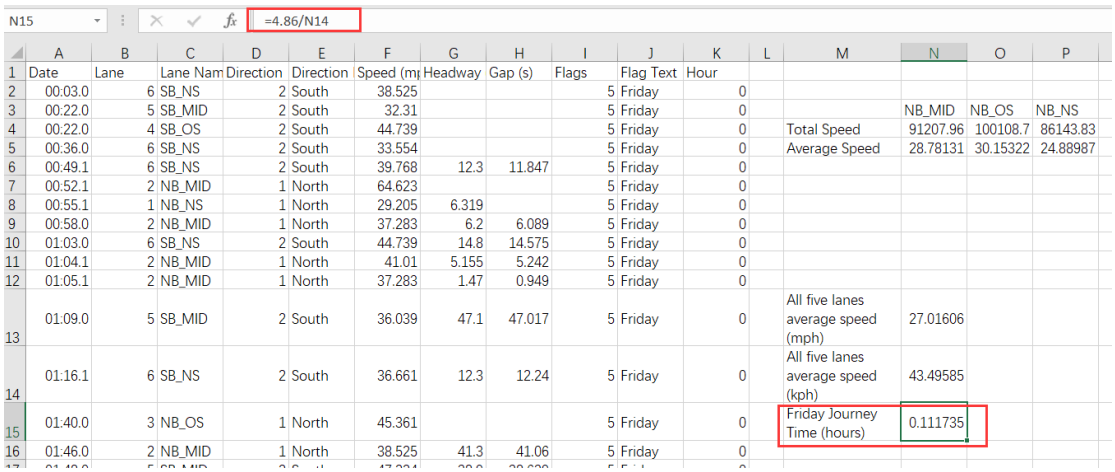

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | Lane | Lane Nam | Direction | Direction | Speed (m | Headway | Gap (s) | Flags | Flag Text | Hour | | | | | |
| 2 | 00:03.0 | 6 | SB_NS | 2 | South | 38.525 | | | 5 | Friday | 0 | | | | | |
| 3 | 00:22.0 | 5 | SB_MID | 2 | South | 32.31 | | | 5 | Friday | 0 | | | NB_MID | NB_OS | NB_NS |
| 4 | 00:22.0 | 4 | SB_OS | 2 | South | 44.739 | | | 5 | Friday | 0 | | Total Speed | 91207.96 | 100108.7 | 86143.83 |
| 5 | 00:36.0 | 6 | SB_NS | 2 | South | 33.554 | | | 5 | Friday | 0 | | Average Speed | 28.78131 | 30.15322 | 24.88987 |
| 6 | 00:49.1 | 6 | SB_NS | 2 | South | 39.768 | 12.3 | 11.847 | 5 | Friday | 0 | | | | | |
| 7 | 00:52.1 | 2 | NB_MID | 1 | North | 64.623 | | | 5 | Friday | 0 | | | | | |
| 8 | 00:55.1 | 1 | NB_NS | 1 | North | 29.205 | 6.319 | | 5 | Friday | 0 | | | | | |
| 9 | 00:58.0 | 2 | NB_MID | 1 | North | 37.283 | 6.2 | 6.089 | 5 | Friday | 0 | | | | | |
| 10 | 01:03.0 | 6 | SB_NS | 2 | South | 44.739 | 14.8 | 14.575 | 5 | Friday | 0 | | | | | |
| 11 | 01:04.1 | 2 | NB_MID | 1 | North | 41.01 | 5.155 | 5.242 | 5 | Friday | 0 | | | | | |
| 12 | 01:05.1 | 2 | NB_MID | 1 | North | 37.283 | 1.47 | 0.949 | 5 | Friday | 0 | | | | | |
| 13 | 01:09.0 | 5 | SB_MID | 2 | South | 36.039 | 47.1 | 47.017 | 5 | Friday | 0 | | All five lanes average speed | 27.01606 | | |
| 14 | 01:16.1 | 6 | SB_NS | 2 | South | 36.661 | 12.3 | 12.24 | 5 | Friday | 0 | | | | | |
| 15 | 01:40.0 | 3 | NB_OS | 1 | North | 45.361 | | | 5 | Friday | 0 | | | | | |
| 16 | 01:46.0 | 2 | NB_MID | 1 | North | 38.525 | 41.3 | 41.06 | 5 | Friday | 0 | | | | | |
| 17 | 01:48.0 | 5 | SB_MID | 2 | South | 47.224 | 38.9 | 38.639 | 5 | Friday | 0 | | | | | |

Convert the average speed to kilometers per hour.



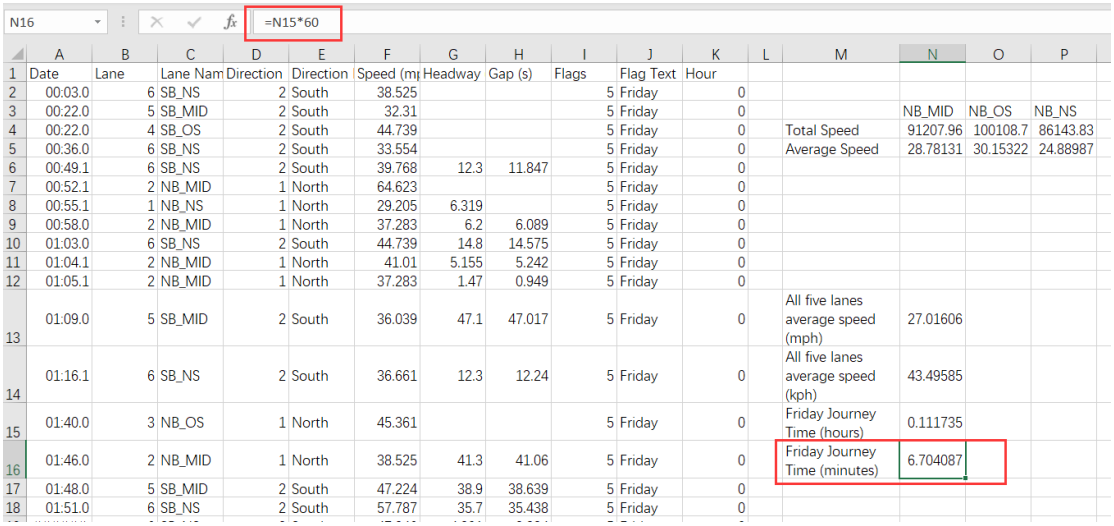

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | Lane | Lane Nam | Direction | Direction | Speed (m | Headway | Gap (s) | Flags | Flag Text | Hour | | | | | |
| 2 | 00:03.0 | 6 | SB_NS | 2 | South | 38.525 | | | 5 | Friday | 0 | | | | | |
| 3 | 00:22.0 | 5 | SB_MID | 2 | South | 32.31 | | | 5 | Friday | 0 | | | NB_MID | NB_OS | NB_NS |
| 4 | 00:22.0 | 4 | SB_OS | 2 | South | 44.739 | | | 5 | Friday | 0 | | Total Speed | 91207.96 | 100108.7 | 86143.83 |
| 5 | 00:36.0 | 6 | SB_NS | 2 | South | 33.554 | | | 5 | Friday | 0 | | Average Speed | 28.78131 | 30.15322 | 24.88987 |
| 6 | 00:49.1 | 6 | SB_NS | 2 | South | 39.768 | 12.3 | 11.847 | 5 | Friday | 0 | | | | | |
| 7 | 00:52.1 | 2 | NB_MID | 1 | North | 64.623 | | | 5 | Friday | 0 | | | | | |
| 8 | 00:55.1 | 1 | NB_NS | 1 | North | 29.205 | 6.319 | | 5 | Friday | 0 | | | | | |
| 9 | 00:58.0 | 2 | NB_MID | 1 | North | 37.283 | 6.2 | 6.089 | 5 | Friday | 0 | | | | | |
| 10 | 01:03.0 | 6 | SB_NS | 2 | South | 44.739 | 14.8 | 14.575 | 5 | Friday | 0 | | | | | |
| 11 | 01:04.1 | 2 | NB_MID | 1 | North | 41.01 | 5.155 | 5.242 | 5 | Friday | 0 | | | | | |
| 12 | 01:05.1 | 2 | NB_MID | 1 | North | 37.283 | 1.47 | 0.949 | 5 | Friday | 0 | | | | | |
| 13 | 01:09.0 | 5 | SB_MID | 2 | South | 36.039 | 47.1 | 47.017 | 5 | Friday | 0 | | All five lanes average speed (mph) | 27.01606 | | |
| 14 | 01:16.1 | 6 | SB_NS | 2 | South | 36.661 | 12.3 | 12.24 | 5 | Friday | 0 | | All five lanes average speed (kph) | 43.49585 | | |
| 15 | 01:40.0 | 3 | NB_OS | 1 | North | 45.361 | | | 5 | Friday | 0 | | | | | |
| 16 | 01:46.0 | 2 | NB_MID | 1 | North | 38.525 | 41.3 | 41.06 | 5 | Friday | 0 | | | | | |

Compute the journey time in hours.




| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | Lane | Lane Nam | Direction | Direction | Speed (m | Headway | Gap (s) | Flags | Flag Text | Hour | | | | | |
| 2 | 00:03.0 | 6 | SB_NS | 2 | South | 38.525 | | | 5 | Friday | 0 | | | | | |
| 3 | 00:22.0 | 5 | SB_MID | 2 | South | 32.31 | | | 5 | Friday | 0 | | | NB_MID | NB_OS | NB_NS |
| 4 | 00:22.0 | 4 | SB_OS | 2 | South | 44.739 | | | 5 | Friday | 0 | | Total Speed | 91207.96 | 100108.7 | 86143.83 |
| 5 | 00:36.0 | 6 | SB_NS | 2 | South | 33.554 | | | 5 | Friday | 0 | | Average Speed | 28.78131 | 30.15322 | 24.88987 |
| 6 | 00:49.1 | 6 | SB_NS | 2 | South | 39.768 | 12.3 | 11.847 | 5 | Friday | 0 | | | | | |
| 7 | 00:52.1 | 2 | NB_MID | 1 | North | 64.623 | | | 5 | Friday | 0 | | | | | |
| 8 | 00:55.1 | 1 | NB_NS | 1 | North | 29.205 | 6.319 | | 5 | Friday | 0 | | | | | |
| 9 | 00:58.0 | 2 | NB_MID | 1 | North | 37.283 | 6.2 | 6.089 | 5 | Friday | 0 | | | | | |
| 10 | 01:03.0 | 6 | SB_NS | 2 | South | 44.739 | 14.8 | 14.575 | 5 | Friday | 0 | | | | | |
| 11 | 01:04.1 | 2 | NB_MID | 1 | North | 41.01 | 5.155 | 5.242 | 5 | Friday | 0 | | | | | |
| 12 | 01:05.1 | 2 | NB_MID | 1 | North | 37.283 | 1.47 | 0.949 | 5 | Friday | 0 | | | | | |
| 13 | 01:09.0 | 5 | SB_MID | 2 | South | 36.039 | 47.1 | 47.017 | 5 | Friday | 0 | | All five lanes average speed (mph) | 27.01606 | | |
| 14 | 01:16.1 | 6 | SB_NS | 2 | South | 36.661 | 12.3 | 12.24 | 5 | Friday | 0 | | All five lanes average speed (kph) | 43.49585 | | |
| 15 | 01:40.0 | 3 | NB_OS | 1 | North | 45.361 | | | 5 | Friday | 0 | | Friday Journey Time (hours) | 0.111735 | | |
| 16 | 01:46.0 | 2 | NB_MID | 1 | North | 38.525 | 41.3 | 41.06 | 5 | Friday | 0 | | | | | |

Convert the journey time to minutes.



| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Date | Lane | Lane Nam | Direction | Direction | Speed (mp | Headway | Gap (s) | Flags | Flag Text | Hour | | | | | |
| 2 | 00:03.0 | 6 | SB_NS | 2 | South | 38.525 | | | 5 | Friday | 0 | | | | | |
| 3 | 00:22.0 | 5 | SB_MID | 2 | South | 32.31 | | | 5 | Friday | 0 | | | NB_MID | NB_OS | NB_NS |
| 4 | 00:22.0 | 4 | SB_OS | 2 | South | 44.739 | | | 5 | Friday | 0 | | Total Speed | 91207.96 | 100108.7 | 86143.83 |
| 5 | 00:36.0 | 6 | SB_NS | 2 | South | 33.554 | | | 5 | Friday | 0 | | Average Speed | 28.78131 | 30.15322 | 24.88987 |
| 6 | 00:49.1 | 6 | SB_NS | 2 | South | 39.768 | 12.3 | 11.847 | 5 | Friday | 0 | | | | | |
| 7 | 00:52.1 | 2 | NB_MID | 1 | North | 64.623 | | | 5 | Friday | 0 | | | | | |
| 8 | 00:55.1 | 1 | NB_NS | 1 | North | 29.205 | 6.319 | | 5 | Friday | 0 | | | | | |
| 9 | 00:58.0 | 2 | NB_MID | 1 | North | 37.283 | 6.2 | 6.089 | 5 | Friday | 0 | | | | | |
| 10 | 01:03.0 | 6 | SB_NS | 2 | South | 44.739 | 14.8 | 14.575 | 5 | Friday | 0 | | | | | |
| 11 | 01:04.1 | 2 | NB_MID | 1 | North | 41.01 | 5.155 | 5.242 | 5 | Friday | 0 | | | | | |
| 12 | 01:05.1 | 2 | NB_MID | 1 | North | 37.283 | 1.47 | 0.949 | 5 | Friday | 0 | | | | | |
| 13 | 01:09.0 | 5 | SB_MID | 2 | South | 36.039 | 47.1 | 47.017 | 5 | Friday | 0 | | All five lanes average speed (mph) | 27.01606 | | |
| 14 | 01:16.1 | 6 | SB_NS | 2 | South | 36.661 | 12.3 | 12.24 | 5 | Friday | 0 | | All five lanes average speed (kph) | 43.49585 | | |
| 15 | 01:40.0 | 3 | NB_OS | 1 | North | 45.361 | | | 5 | Friday | 0 | | Friday Journey Time (hours) | 0.111735 | | |
| 16 | 01:46.0 | 2 | NB_MID | 1 | North | 38.525 | 41.3 | 41.06 | 5 | Friday | 0 | | Friday Journey Time (minutes) | 6.704087 | | |
| 17 | 01:48.0 | 5 | SB_MID | 2 | South | 47.224 | 38.9 | 38.639 | 5 | Friday | 0 | | | | | |
| 18 | 01:51.0 | 6 | SB_NS | 2 | South | 57.787 | 35.7 | 35.438 | 5 | Friday | 0 | | | | | |

***Task 7.2 Result (2 Decimal):*** Journey time is 6.704 minutes

## Python and Excel Comparison

### Similarity

They can all be used for further processing, analysis and visualization of the data. Both have functions already defined and can be used directly. Both can operate on csv files.

### Python

Advantage:

1. In addition to some self-contained functions, it is easy to download third-party libraries to make data processing simpler and more diverse (suck as machine learning).
2. More efficient when handling large amounts of data.
3. By writing the code once, repetitive operations can be performed.
4. The code is readable and can be saved, so it can be disseminated and improved.
5. It can be used in multiple platforms.
6. Can read and process many types of files

Disadvantage

1. Relatively difficult to learn.
2. May face various bugs.

### Excel

Advantage

1. It has a graphical interface. Process data with a mouse click without writing a lot of code. More intuitive operation.
2. Easy to learn, excel can be used by short learning time.

3. High penetration rate, most computers have Excel.

Disadvantage

1. It gets laggy when processing large amounts of data
2. When performing the same operation on data, the operation needs to be repeated.
3. The process is not recorded after processing.

For this task, I prefer Python to Excel. Because Excel has too many repetitive operations. For instance, when calculating average speed of each lane, it uses same function repeatedly by just changing a condition. Python can easily do this once by using groupby() function. The running time of both two technology are in small difference.