DATA CLEANING

IMPORTING LIBRARIES

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

LOADING CSV FILE

In [2]:
```
weather=pd.read_csv('F:/PYTHON/Refactored_Py_DS_ML_Bootcamp-master/weather.csv', encoding='latin1')
```

CHECKING FIRST FIVE ITEMS/ ROWS OF DATA SET

In [3]:
```
weather.head()
```

Out[3]:

	Date/Time	Year	Month	Day	Data Quality	Max Temp (°C)	Max Temp Flag	Min Temp (°C)	Min Temp Flag	Mean Temp (°C)	...	Total Snow (cm)	Total Snow Flag	F
0	01-01-17	2017	1	1	NaN	3.1	NaN	-0.5	NaN	1.3	...	NaN	M	
1	02-01-17	2017	1	2	NaN	5.5	NaN	0.7	NaN	3.1	...	NaN	M	
2	03-01-17	2017	1	3	NaN	4.8	NaN	2.4	NaN	3.6	...	NaN	M	
3	04-01-17	2017	1	4	NaN	3.8	NaN	-7.6	NaN	-1.9	...	NaN	M	
4	05-01-17	2017	1	5	NaN	-5.3	NaN	-10.3	NaN	-7.8	...	NaN	M	

5 rows × 27 columns

CHECKING FOR "NA" VALUES

In [4]:
```
weather.isna().head()
```

Out[4]:

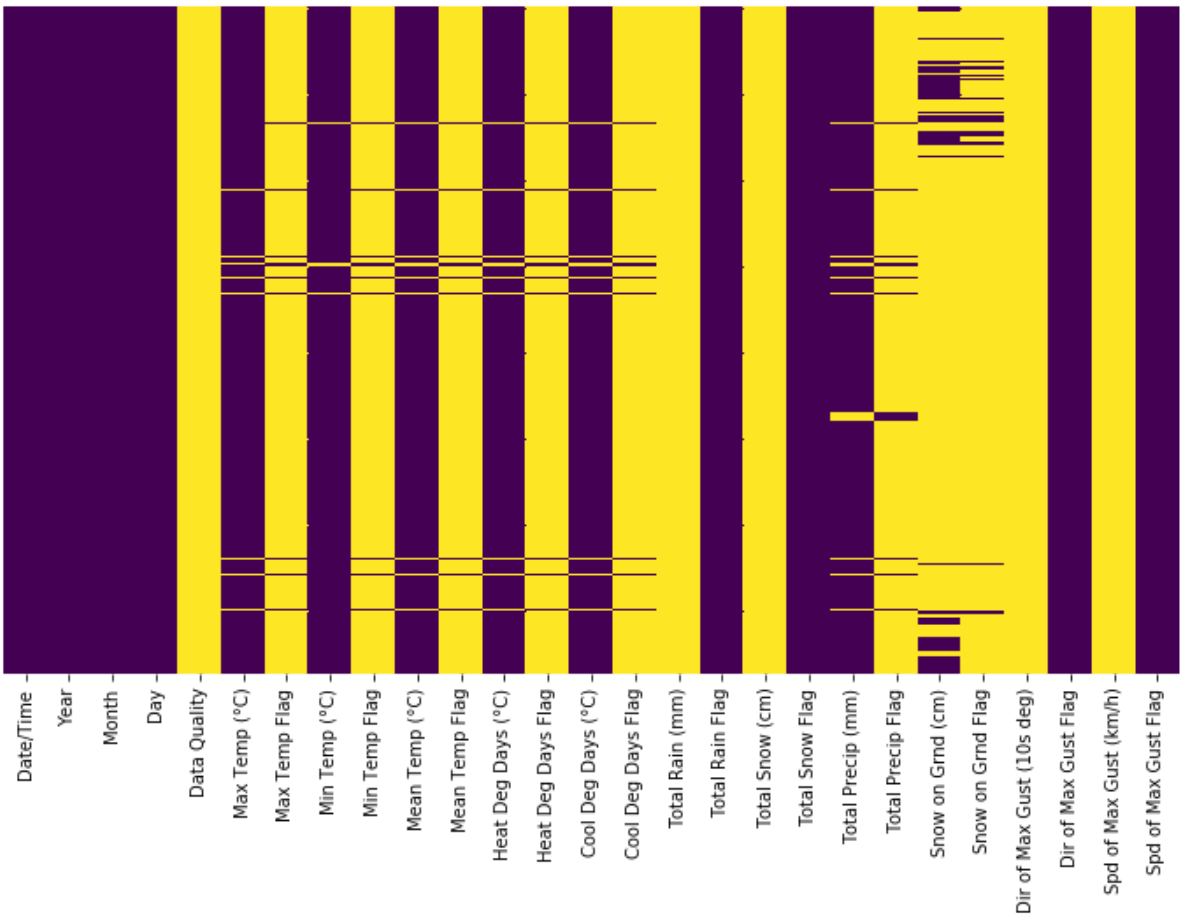
	Date/Time	Year	Month	Day	Data Quality	Max Temp (°C)	Max Temp Flag	Min Temp (°C)	Min Temp Flag	Mean Temp (°C)	...	Total Snow (cm)	Total Snow Flag
0	False	False	False	False	True	False	True	False	True	False	...	True	False
1	False	False	False	False	True	False	True	False	True	False	...	True	False
2	False	False	False	False	True	False	True	False	True	False	...	True	False
3	False	False	False	False	True	False	True	False	True	False	...	True	False
4	False	False	False	False	True	False	True	False	True	False	...	True	False

5 rows × 27 columns

EXPLORATORY ANALYSIS

In [5]:
```
plt.figure(figsize=(12, 7))
sns.heatmap(weather.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[5]: <matplotlib.axes._subplots.AxesSubplot at 0x897b14828>



CALCULATING NUMBER OF ITEMS IN A EVERY COLUMN

In [6]:
```
weather.isnull().values.any()
```

Out[6]: True

In [7]:
```
weather.isnull().sum().sum()
```

Out[7]: 4657

In [8]:
```
weather.count()
```

Out[8]:
```
Date/Time                   365
Year                        365
Month                       365
Day                         365
Data Quality                  0
Max Temp (°C)               357
Max Temp Flag                10
Min Temp (°C)               361
Min Temp Flag                10
Mean Temp (°C)              355
Mean Temp Flag               10
Heat Deg Days (°C)          355
Heat Deg Days Flag           10
Cool Deg Days (°C)          355
Cool Deg Days Flag           10
Total Rain (mm)               0
Total Rain Flag             365
Total Snow (cm)               0
Total Snow Flag             365
Total Precip (mm)           350
Total Precip Flag            15
Snow on Grnd (cm)            60
Snow on Grnd Flag            20
Dir of Max Gust (10s deg)     0
Dir of Max Gust Flag        365
Spd of Max Gust (km/h)        0
Spd of Max Gust Flag        365
dtype: int64
```

In [9]:
```
weather.isnull().sum()
```

Out[9]:
```
Date/Time                    0
Year                         0
Month                        0
Day                          0
Data Quality               365
Max Temp (°C)                8
Max Temp Flag              355
Min Temp (°C)                4
Min Temp Flag              355
Mean Temp (°C)              10
Mean Temp Flag             355
Heat Deg Days (°C)          10
Heat Deg Days Flag         355
Cool Deg Days (°C)          10
Cool Deg Days Flag         355
Total Rain (mm)            365
Total Rain Flag              0
Total Snow (cm)            365
Total Snow Flag              0
Total Precip (mm)           15
Total Precip Flag          350
Snow on Grnd (cm)          305
Snow on Grnd Flag          345
Dir of Max Gust (10s deg)  365
Dir of Max Gust Flag         0
Spd of Max Gust (km/h)     365
Spd of Max Gust Flag         0
dtype: int64
```

DROPPING EMPTY COLUMNS

In [10]:
```
to_drop = ['Dir of Max Gust (10s deg)', 'Total Snow (cm)', 'Total Rain (mm)']
weather.drop(to_drop, inplace=True, axis=1)
to_drop = ['Spd of Max Gust (km/h)']
weather.drop(to_drop, inplace=True, axis=1)
weather.head()
```

Out[10]:

	Date/Time	Year	Month	Day	Data Quality	Max Temp (°C)	Max Temp Flag	Min Temp (°C)	Min Temp Flag	Mean Temp (°C)	...	Cool Deg Days (°C)	Cool Deg Days Flag	To R F
0	01-01-17	2017	1	1	NaN	3.1	NaN	-0.5	NaN	1.3	...	0.0	NaN	
1	02-01-17	2017	1	2	NaN	5.5	NaN	0.7	NaN	3.1	...	0.0	NaN	
2	03-01-17	2017	1	3	NaN	4.8	NaN	2.4	NaN	3.6	...	0.0	NaN	
3	04-01-17	2017	1	4	NaN	3.8	NaN	-7.6	NaN	-1.9	...	0.0	NaN	
4	05-01-17	2017	1	5	NaN	-5.3	NaN	-10.3	NaN	-7.8	...	0.0	NaN	

5 rows × 23 columns

In [11]:
```
to_drop = ['Data Quality']
weather.drop(to_drop, inplace=True, axis=1)
weather.head()
```

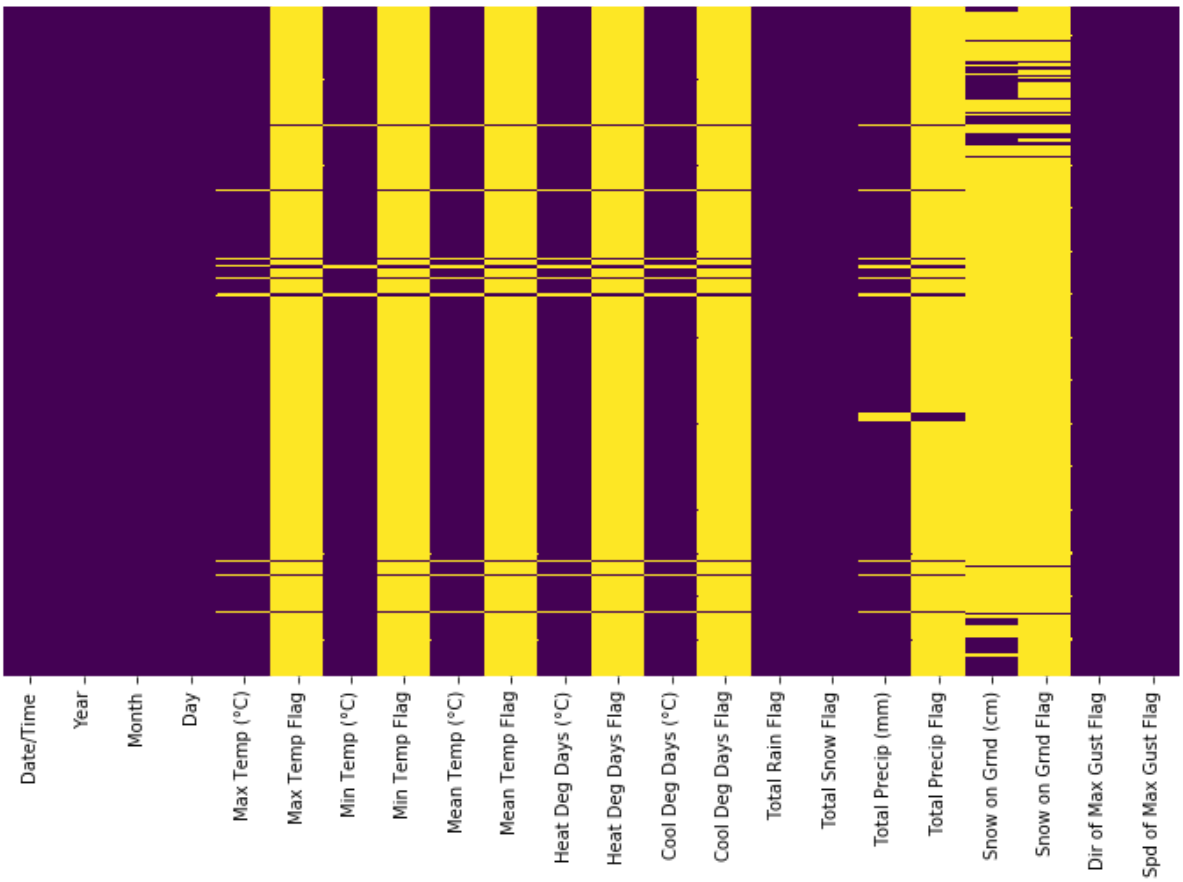
Out[11]:

	Date/Time	Year	Month	Day	Max Temp (°C)	Max Temp Flag	Min Temp (°C)	Min Temp Flag	Mean Temp (°C)	Mean Temp Flag	...	Cool Deg Days (°C)	Cool Deg Days Flag	Tot Rai Fla
0	01-01-17	2017	1	1	3.1	NaN	-0.5	NaN	1.3	NaN	...	0.0	NaN	
1	02-01-17	2017	1	2	5.5	NaN	0.7	NaN	3.1	NaN	...	0.0	NaN	
2	03-01-17	2017	1	3	4.8	NaN	2.4	NaN	3.6	NaN	...	0.0	NaN	
3	04-01-17	2017	1	4	3.8	NaN	-7.6	NaN	-1.9	NaN	...	0.0	NaN	
4	05-01-17	2017	1	5	-5.3	NaN	-10.3	NaN	-7.8	NaN	...	0.0	NaN	

5 rows × 22 columns

In [12]:
```
plt.figure(figsize=(12, 7))
sns.heatmap(weather.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x8985c3b00>



COMPLETING FLAGS

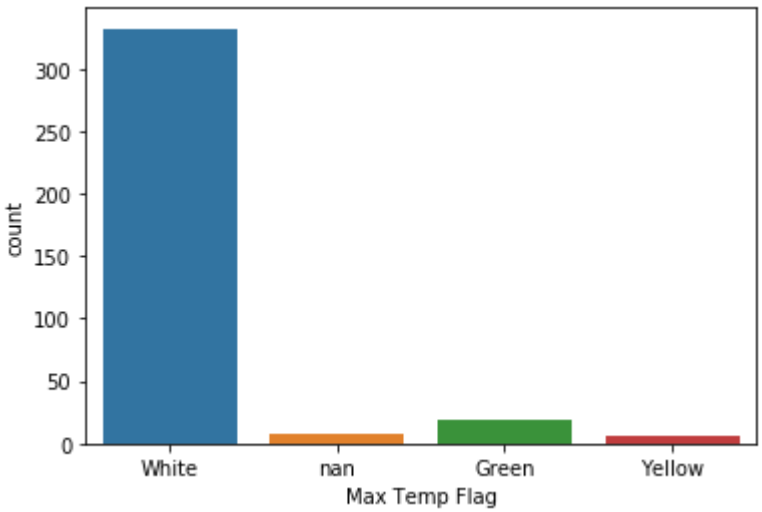
REFERENCE FOR FLAGS: https://learn.weatherstem.com/modules/learn/lessons/27/12.html (https://learn.weatherstem.com/modules/learn/lessons/27/12.html)

In [13]:
```
#SIngle condition : weather['Max Temp Flag'] = np.where(weather['Max Temp (°C)'] >33.3, 'Green',weather['Max Temp Flag'])

conditions  = [ weather['Max Temp (°C)'] >33.3, (weather['Max Temp (°C)'] <=33.3) & (weather['Max Temp (°C)'] >=32.3), (weather['Max Temp (°C)'] <32.3) & (weather['Max Temp (°C)'] >=30.5), (weather['Max Temp (°C)'] <30.5) & (weather['Max Temp (°C)'] >=27.8), weather['Max Temp (°C)'] <27.8 ]
choices     = [ "Black", 'Red', 'Yellow', 'Green', 'White']

weather['Max Temp Flag'] = np.select(conditions, choices, default=np.nan)

weather['Max Temp Flag'].value_counts()
```

Out[13]:
```
White     332
Green      19
nan         8
Yellow      6
Name: Max Temp Flag, dtype: int64
```

In [14]:
```
sns.countplot(x='Max Temp Flag', data=weather)
```

Out[14]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x8985adb00>
```



In [15]:
```
conditions  = [ weather['Min Temp (°C)'] >33.3, (weather['Min Temp (°C)'] <=33.3) & (weather['Min Temp (°C)'] >=32.3), (weather['Min Temp (°C)'] <32.3) & (weather['Min Temp (°C)'] >=30.5), (weather['Min Temp (°C)'] <30.5) & (weather['Min Temp (°C)'] >=27.8), weather['Min Temp (°C)'] <27.8 ]
choices     = [ "Black", 'Red', 'Yellow', 'Green', 'White']

weather['Min Temp Flag'] = np.select(conditions, choices, default=np.nan)

weather['Min Temp Flag'].value_counts()
```

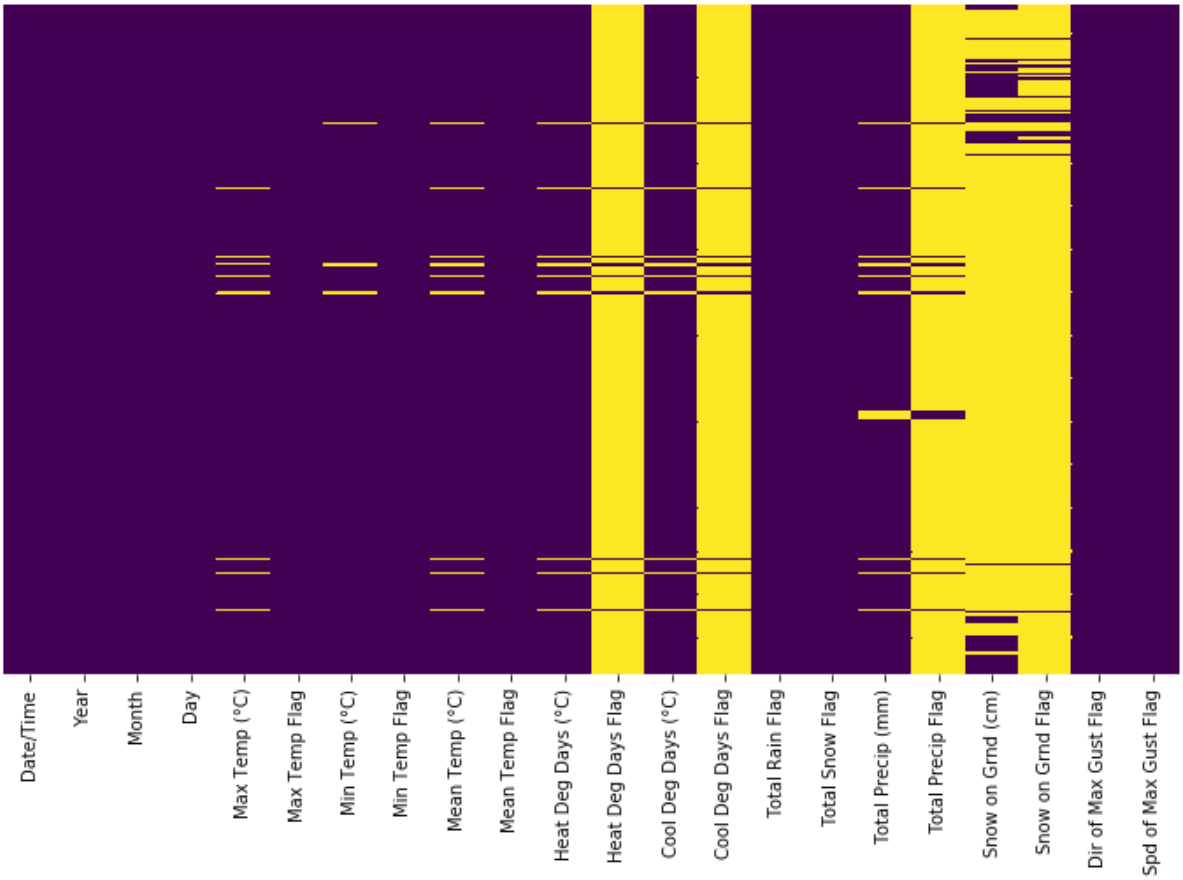
Out[15]:
```
White    361
nan        4
Name: Min Temp Flag, dtype: int64
```

In [16]:
```
conditions  = [ weather['Mean Temp (°C)'] >33.3, (weather['Mean Temp (°C)'] <=
33.3) & (weather['Mean Temp (°C)'] >=32.3), (weather['Mean Temp (°C)'] <32.3)
& (weather['Mean Temp (°C)'] >=30.5), (weather['Mean Temp (°C)'] <30.5) & (wea
ther['Mean Temp (°C)'] >=27.8), weather['Mean Temp (°C)'] <27.8 ]
choices     = [ "Black", 'Red', 'Yellow', 'Green', 'White']

weather['Mean Temp Flag'] = np.select(conditions, choices, default=np.nan)

weather['Mean Temp Flag'].value_counts()
```

Out[16]:
```
White    355
nan       10
Name: Mean Temp Flag, dtype: int64
```

In [17]:
```
plt.figure(figsize=(12, 7))
sns.heatmap(weather.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[17]: <matplotlib.axes._subplots.AxesSubplot at 0x8984c3390>



CHECKING WHICH ROWS HAS NULL VALUES CORRESPONDING TO COLUMNS

In [18]:
```
weather[weather['Max Temp (°C)'].isnull()].index.tolist()
```

Out[18]: [100, 137, 141, 148, 157, 302, 310, 330]

In [19]:
```
weather[weather['Min Temp (°C)'].isnull()].index.tolist()
```

Out[19]:
```
[64, 141, 142, 157]
```

In [20]:
```
weather[weather['Mean Temp (°C)'].isnull()].index.tolist()
```

Out[20]:
```
[64, 100, 137, 141, 142, 148, 157, 302, 310, 330]
```

DROPPING ROWS 141 and 157 AS THESE ROWS ARE EMPTY IN 3 COLUMNS

In [21]:
```
weather.drop(weather.index[[141,157]]).head()
```

Out[21]:

	Date/Time	Year	Month	Day	Max Temp (°C)	Max Temp Flag	Min Temp (°C)	Min Temp Flag	Mean Temp (°C)	Mean Temp Flag	...	Cool Deg Days (°C)	Cool Deg Days Flag	Tot Rai Fla
0	01-01-17	2017	1	1	3.1	White	-0.5	White	1.3	White	...	0.0	NaN	
1	02-01-17	2017	1	2	5.5	White	0.7	White	3.1	White	...	0.0	NaN	
2	03-01-17	2017	1	3	4.8	White	2.4	White	3.6	White	...	0.0	NaN	
3	04-01-17	2017	1	4	3.8	White	-7.6	White	-1.9	White	...	0.0	NaN	
4	05-01-17	2017	1	5	-5.3	White	-10.3	White	-7.8	White	...	0.0	NaN	

5 rows × 22 columns

COOLING DEGREE DAYS

Cooling degree-days for a given day are the number of degrees Celsius that the mean temperature is above 18 °C. If the temperature is equal to or less than 18 °C, then the number will be zero. For example, a day with a mean temperature of 20.5 °C has 2.5 cooling degree-days; a day with a mean temperature of 15.5 °C has zero cooling degree-days. Cooling degree-days are used primarily to estimate the air-conditioning requirements of buildings. REFERENCE: https://climate.weather.gc.ca/glossary_e.html (https://climate.weather.gc.ca/glossary_e.html) accessed on: 26-Sep 2019

There is no information on CDD or HDD flags. So, drop these columns.

In [22]:
```
to_drop = ['Cool Deg Days Flag', 'Heat Deg Days Flag']
weather.drop(to_drop, inplace=True, axis=1)
```

In [23]:
```
weather.columns
```

Out[23]:
```
Index(['Date/Time', 'Year', 'Month', 'Day', 'Max Temp (°C)', 'Max Temp Flag',
       'Min Temp (°C)', 'Min Temp Flag', 'Mean Temp (°C)', 'Mean Temp Flag',
       'Heat Deg Days (°C)', 'Cool Deg Days (°C)', 'Total Rain Flag',
       'Total Snow Flag', 'Total Precip (mm)', 'Total Precip Flag',
       'Snow on Grnd (cm)', 'Snow on Grnd Flag', 'Dir of Max Gust Flag',
       'Spd of Max Gust Flag'],
      dtype='object')
```

DROPPING OTHER FLAG COLUMNS WHICH ARE NOT REQUIRED

In [24]:
```
to_drop = ['Total Rain Flag', 'Total Snow Flag']
weather.drop(to_drop, inplace=True, axis=1)
```

In [25]:
```
weather.columns
```

Out[25]:
```
Index(['Date/Time', 'Year', 'Month', 'Day', 'Max Temp (°C)', 'Max Temp Flag',
       'Min Temp (°C)', 'Min Temp Flag', 'Mean Temp (°C)', 'Mean Temp Flag',
       'Heat Deg Days (°C)', 'Cool Deg Days (°C)', 'Total Precip (mm)',
       'Total Precip Flag', 'Snow on Grnd (cm)', 'Snow on Grnd Flag',
       'Dir of Max Gust Flag', 'Spd of Max Gust Flag'],
      dtype='object')
```

In [26]:
```
weather[weather['Heat Deg Days (°C)'].isna()]
```

Out[26]:

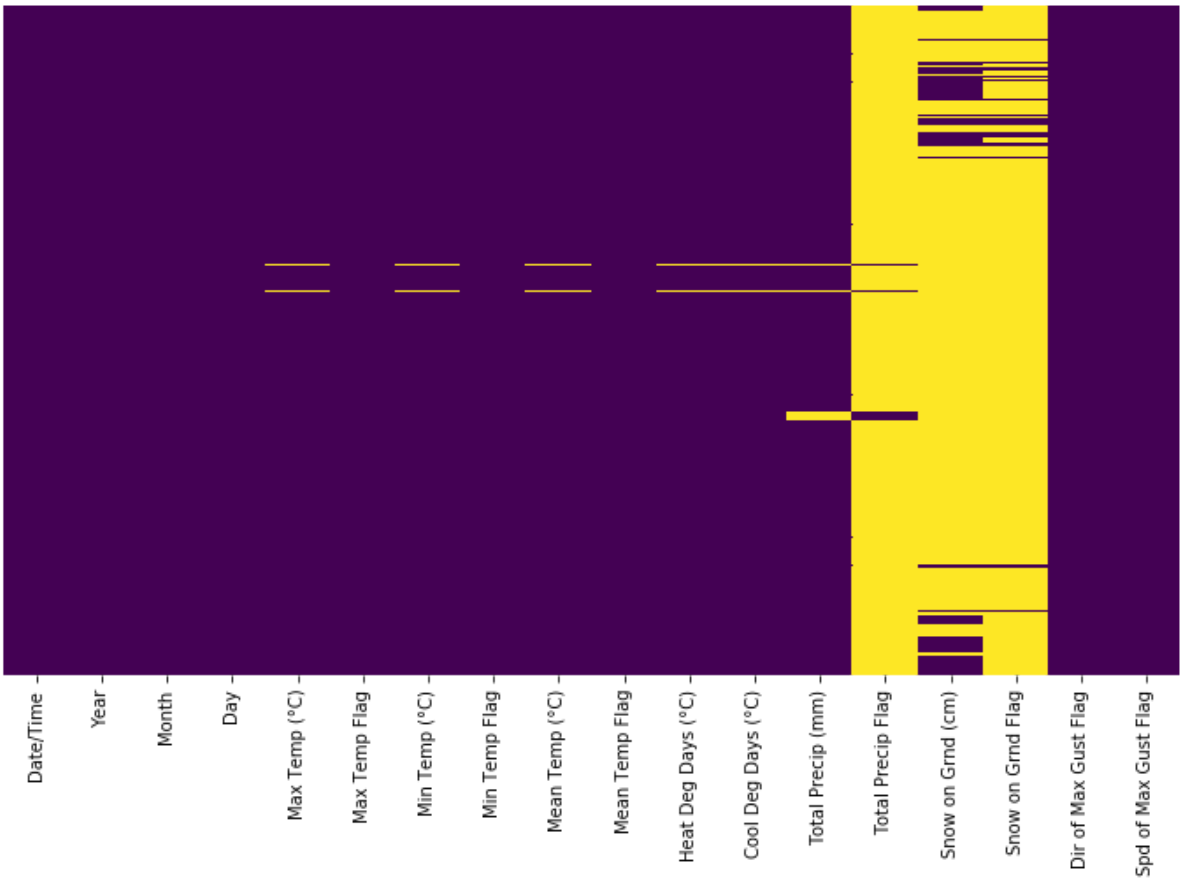
	Date/Time	Year	Month	Day	Max Temp (°C)	Max Temp Flag	Min Temp (°C)	Min Temp Flag	Mean Temp (°C)	Mean Temp Flag	Heat Deg Days (°C)	Cool Deg Days (°C)	Tot Preci (mn
64	06-03-17	2017	3	6	4.9	White	NaN	nan	NaN	nan	NaN	NaN	Na
100	11-04-17	2017	4	11	NaN	nan	11.3	White	NaN	nan	NaN	NaN	Na
137	18-05-17	2017	5	18	NaN	nan	18.4	White	NaN	nan	NaN	NaN	Na
141	22-05-17	2017	5	22	NaN	nan	NaN	nan	NaN	nan	NaN	NaN	Na
142	23-05-17	2017	5	23	21.8	White	NaN	nan	NaN	nan	NaN	NaN	Na
148	29-05-17	2017	5	29	NaN	nan	14.7	White	NaN	nan	NaN	NaN	Na
157	07-06-17	2017	6	7	NaN	nan	NaN	nan	NaN	nan	NaN	NaN	Na
302	30-10-17	2017	10	30	NaN	nan	5.9	White	NaN	nan	NaN	NaN	Na
310	07-11-17	2017	11	7	NaN	nan	4.0	White	NaN	nan	NaN	NaN	Na
330	27-11-17	2017	11	27	NaN	nan	1.3	White	NaN	nan	NaN	NaN	Na

Above output shows that Heat Degree Days has null values where almost all columns are unknown, except few. The available values in these rows only for Max or Min Temp, even not together. So, finding values for all coulmns based on these few values is not certain. So drop these rows.

In [27]:
```
weather.drop([64,100,137,142,148,302,310,330], inplace= True)
```

In [28]:
```
plt.figure(figsize=(12, 7))
sns.heatmap(weather.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[28]: <matplotlib.axes._subplots.AxesSubplot at 0x8985534e0>



DEALING WITH PRECIPIATION FLAG

The millimetre (mm) is the unit of measurement of liquid precipitation and the vertical depth of water or water equivalent is express to the nearest 0.2 mm. Less than 0.2 mm is called a "Trace". https://climate.weather.gc.ca/glossary_e.html (https://climate.weather.gc.ca/glossary_e.html) accessed on: 26 Sep 2019

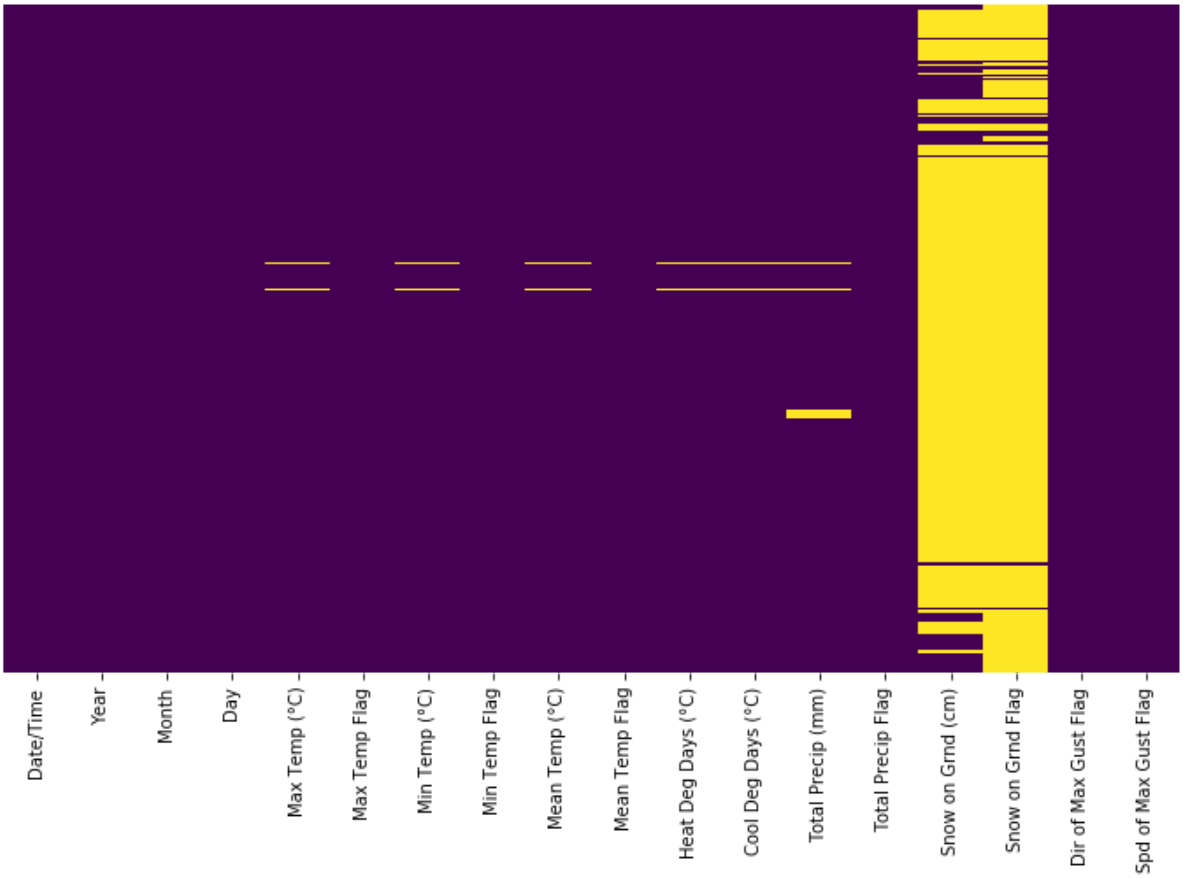
In [29]:
```
conditions  = [ weather['Total Precip (mm)']<0.2, weather['Total Precip (mm)']
>=0.2]
choices     = [ '0', '1'] # 0 is for trace, 1 is for above trace

weather['Total Precip Flag'] = np.select(conditions, choices, default=np.nan)
```

In [30]:
```
weather['Total Precip Flag'].value_counts()
```

Out[30]:
```
0      210
1      140
nan      7
Name: Total Precip Flag, dtype: int64
```

In [31]:
```
plt.figure(figsize=(12, 7))
sns.heatmap(weather.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[31]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x89858cb38>
```

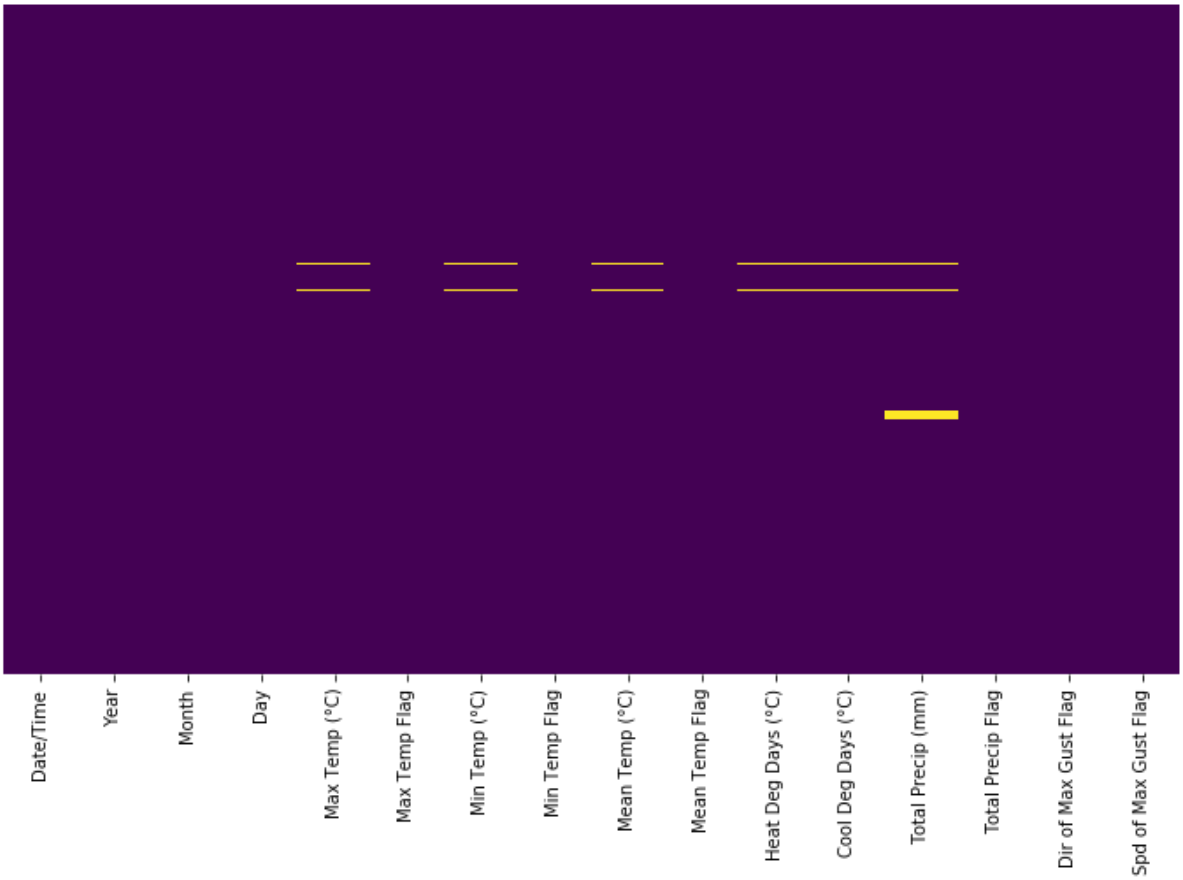


Well, most of the data is clean. However, still there are some missing values which could be cleaned in similar manner. This could be cleaned according to business requirement.

In [32]:
```
to_drop = ['Snow on Grnd (cm)', 'Snow on Grnd Flag']
weather.drop(to_drop, inplace=True, axis=1)
```

In [33]:
```
plt.figure(figsize=(12, 7))
sns.heatmap(weather.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[33]: <matplotlib.axes._subplots.AxesSubplot at 0x898c2ef98>



In [34]:
```
weather[weather['Total Precip (mm)'].isnull()].index.tolist()
```

Out[34]: [141, 157, 222, 223, 224, 225, 226]

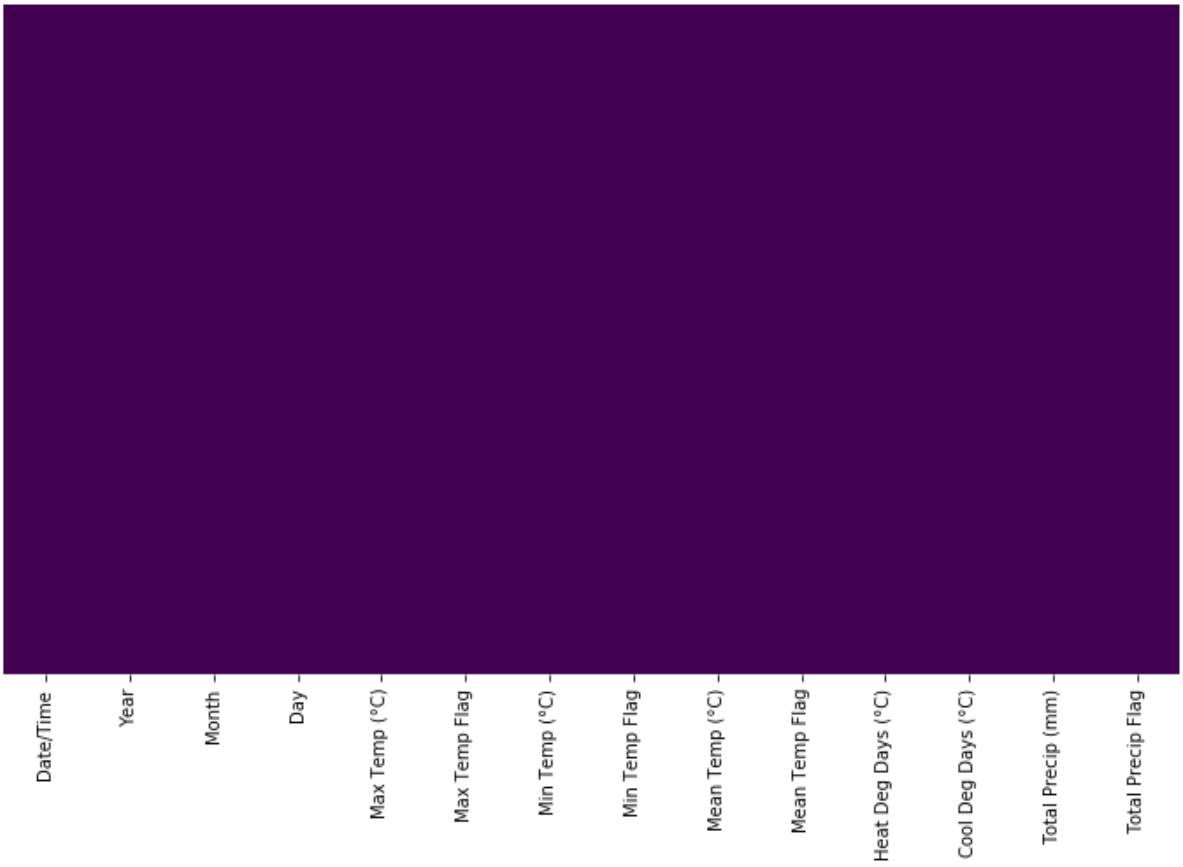
In [35]:
```
weather.drop([141,157,222,223,224,225,226], inplace= True)
```

In [36]:
```
to_drop = ['Dir of Max Gust Flag', 'Spd of Max Gust Flag']
weather.drop(to_drop, inplace=True, axis=1)
```

In [37]:
```
export_csv = weather.to_csv (r'F:\export_weather.csv', index = None, header=True)
```

In [38]:
```
plt.figure(figsize=(12, 7))
sns.heatmap(weather.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[38]: <matplotlib.axes._subplots.AxesSubplot at 0x898ca6160>



DATA CLEANSING, ANALYSIS AND PREPARATION IS COMPLETED!

In [ ]:
```
```