In [442]:
```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns

from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.seasonal import seasonal_decompose
from pandas.plotting import autocorrelation_plot

import math
import calendar
```

In [443]:
```
data_folder = "data/"

data_dengue = "ph_dengue_C3.csv"
df_dengue = pd.read_csv(data_folder + data_dengue)

data_climate = "ph_climate_weekly_C3_.csv"
df_climate = pd.read_csv(data_folder + data_climate)

data_coord = "ph_cities_coordinates_C3.csv"
df_coord = pd.read_csv(data_folder + data_coord)

data_trend_01 = "ph_google trend__dengue symptoms_C3.csv"
df_trend_01 = pd.read_csv(data_folder + data_trend_01)

data_trend_02 = "ph_google trend__dengue_C3.csv"
df_trend_02 = pd.read_csv(data_folder + data_trend_02)

data_hospital = "ph_hospitals_C3.csv"
df_hospital = pd.read_csv(data_folder + data_hospital)

data_static = "ph_static_C3.csv"
df_static = pd.read_csv(data_folder + data_static)
```

In [444]:
```
plt.style.use('seaborn-whitegrid')
```

Dataframe Checking

df_dengue

In [445]:
```
df_dengue.head()
```

Out[445]:

	loc	cases	deaths	date	Region	year	month
0	BULACAN	97	0	2016-01-10	REGION III-CENTRAL LUZON	2016	1
1	BULACAN	58	0	2016-01-17	REGION III-CENTRAL LUZON	2016	1
2	BULACAN	76	1	2016-01-24	REGION III-CENTRAL LUZON	2016	1
3	BULACAN	101	0	2016-01-31	REGION III-CENTRAL LUZON	2016	1
4	BULACAN	85	0	2016-02-07	REGION III-CENTRAL LUZON	2016	2

In [446]:
```
df_dengue['date'] = pd.to_datetime(df_dengue['date'])
print(f"Top 3 Max Date:\n{df_dengue['date'].drop_duplicates().nlargest(3)}")
print(f"Top 3 Min Date:\n{df_dengue['date'].drop_duplicates().nsmallest(3)}")
```

```
Top 3 Max Date:
675   2021-01-10
674   2021-01-03
673   2020-12-27
Name: date, dtype: datetime64[ns]
Top 3 Min Date:
0   2016-01-10
1   2016-01-17
2   2016-01-24
Name: date, dtype: datetime64[ns]
```

In [447]:
```
dates_to_remove = ['2021-01-10', '2021-01-03']
mask = ~df_dengue['date'].isin(dates_to_remove)
df_dengue = df_dengue[mask]
```

In [448]:
```
print(f"Max Date: {df_dengue['date'].max()}")
print(f"Min Date: {df_dengue['date'].min()}")
```

```
Max Date: 2020-12-27 00:00:00
Min Date: 2016-01-10 00:00:00
```

In [449]:
```
print(len(df_dengue))
```

```
775
```

In [450]:
```
df_dengue = df_dengue.rename(columns={"Region": "region"})
df_dengue = df_dengue[["date","loc","region","cases","deaths"]]
df_dengue.head()
```

Out[450]:

	date	loc	region	cases	deaths
0	2016-01-10	BULACAN	REGION III-CENTRAL LUZON	97	0
1	2016-01-17	BULACAN	REGION III-CENTRAL LUZON	58	0
2	2016-01-24	BULACAN	REGION III-CENTRAL LUZON	76	1
3	2016-01-31	BULACAN	REGION III-CENTRAL LUZON	101	0
4	2016-02-07	BULACAN	REGION III-CENTRAL LUZON	85	0

In [451]:
```
print(f"Date Range: {str(df_dengue['date'].min())} to {str(df_dengue['date'].max())}")
for loc, group in df_dengue.groupby('loc'):
    region = group['region'].iloc[0]
    print(f"Location: {loc}, Region: {region}")
```

```
Date Range: 2016-01-10 00:00:00 to 2020-12-27 00:00:00
Location: BULACAN, Region: REGION III-CENTRAL LUZON
Location: QUEZON CITY, Region: NATIONAL CAPITAL REGION
Location: RIZAL, Region: REGION IV-A-CALABARZON
```

In [452]:
```
df_dengue["date"] = pd.to_datetime(df_dengue["date"])
df_dengue = df_dengue.sort_values("date")
df_dengue.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 775 entries, 0 to 773
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype         
---  ------  --------------  -----         
 0   date    775 non-null    datetime64[ns]
 1   loc     775 non-null    object        
 2   region  775 non-null    object        
 3   cases   775 non-null    int64         
 4   deaths  775 non-null    int64         
dtypes: datetime64[ns](1), int64(2), object(2)
memory usage: 36.3+ KB
```

In [453]:
```
print(f"Number of Duplicated Rows: {df_dengue.duplicated().sum()}")
```

```
Number of Duplicated Rows: 0
```

In [454]:
```
df_dengue.describe()
```

Out[454]:

	cases	deaths
count	775.000000	775.000000
mean	120.912258	1.952258
std	123.439029	13.669466
min	0.000000	0.000000
25%	39.000000	0.000000
50%	83.000000	0.000000
75%	157.000000	1.000000
max	812.000000	203.000000

In [455]:
```
df_dengue_bulacan = df_dengue[df_dengue["loc"] == "BULACAN"]
df_dengue_quezon = df_dengue[df_dengue["loc"] == "QUEZON CITY"]
df_dengue_rizal = df_dengue[df_dengue["loc"] == "RIZAL"]
```

In [456]:
```
df_dengue_bulacan = df_dengue_bulacan.set_index("date")
df_dengue_bulacan = df_dengue_bulacan.asfreq("W")
df_dengue_bulacan.head()
```

Out[456]:

date	loc	region	cases	deaths
2016-01-10	BULACAN	REGION III-CENTRAL LUZON	97.0	0.0
2016-01-17	BULACAN	REGION III-CENTRAL LUZON	58.0	0.0
2016-01-24	BULACAN	REGION III-CENTRAL LUZON	76.0	1.0
2016-01-31	BULACAN	REGION III-CENTRAL LUZON	101.0	0.0
2016-02-07	BULACAN	REGION III-CENTRAL LUZON	85.0	0.0

In [457]:
```
df_dengue_quezon = df_dengue_quezon.set_index("date")
df_dengue_quezon = df_dengue_quezon.asfreq("W")
df_dengue_quezon.head()
```

Out[457]:

date	loc	region	cases	deaths
2016-01-10	QUEZON CITY	NATIONAL CAPITAL REGION	58.0	0.0
2016-01-17	QUEZON CITY	NATIONAL CAPITAL REGION	81.0	0.0
2016-01-24	QUEZON CITY	NATIONAL CAPITAL REGION	85.0	0.0
2016-01-31	QUEZON CITY	NATIONAL CAPITAL REGION	73.0	1.0
2016-02-07	QUEZON CITY	NATIONAL CAPITAL REGION	75.0	2.0

In [458]:
```
df_dengue_rizal = df_dengue_rizal.set_index("date")
df_dengue_rizal = df_dengue_rizal.asfreq("W")
df_dengue_rizal.head()
```

Out[458]:

date	loc	region	cases	deaths
2016-01-10	RIZAL	REGION IV-A-CALABARZON	93.0	0.0
2016-01-17	RIZAL	REGION IV-A-CALABARZON	78.0	3.0
2016-01-24	RIZAL	REGION IV-A-CALABARZON	111.0	2.0
2016-01-31	RIZAL	REGION IV-A-CALABARZON	109.0	0.0
2016-02-07	RIZAL	REGION IV-A-CALABARZON	98.0	2.0

In [459]:
```
print(f"Number of Null Values in Bulacan df: \n{df_dengue_bulacan.isnull().sum()}\n")
print(f"Number of Null Values in Quezon df: \n{df_dengue_quezon.isnull().sum()}\n")
print(f"Number of Null Values in Rizal df: \n{df_dengue_rizal.isnull().sum()}")
```

```
Number of Null Values in Bulacan df:
loc       1
region    1
cases     1
deaths    1
dtype: int64

Number of Null Values in Quezon df:
loc       3
region    3
cases     3
deaths    3
dtype: int64

Number of Null Values in Rizal df:
loc       1
region    1
cases     1
deaths    1
dtype: int64
```

In [460]:
```
print(f"Null Values in Bulacan df: \n{df_dengue_bulacan[df_dengue_bulacan.isnull().any(axis=1)]}\n")
print(f"Null Values in Quezon df: \n{df_dengue_quezon[df_dengue_quezon.isnull().any(axis=1)]}\n")
print(f"Null Values in Rizal df: \n{df_dengue_rizal[df_dengue_rizal.isnull().any(axis=1)]}")
```

```
Null Values in Bulacan df:
            loc region  cases  deaths
date
2019-01-06  NaN    NaN    NaN     NaN

Null Values in Quezon df:
            loc region  cases  deaths
date
2016-11-13  NaN    NaN    NaN     NaN
2019-01-06  NaN    NaN    NaN     NaN
2020-11-15  NaN    NaN    NaN     NaN

Null Values in Rizal df:
            loc region  cases  deaths
date
2019-01-06  NaN    NaN    NaN     NaN
```

In [461]:
```
df_dengue_bulacan = df_dengue_bulacan.fillna(method='ffill')
df_dengue_quezon = df_dengue_quezon.fillna(method='ffill')
df_dengue_rizal = df_dengue_rizal.fillna(method='ffill')

print(f"Number of Null Values in Bulacan df: \n{df_dengue_bulacan.isnull().sum()}\n")
print(f"Number of Null Values in Quezon df: \n{df_dengue_quezon.isnull().sum()}\n")
print(f"Number of Null Values in Rizal df: \n{df_dengue_rizal.isnull().sum()}")
```

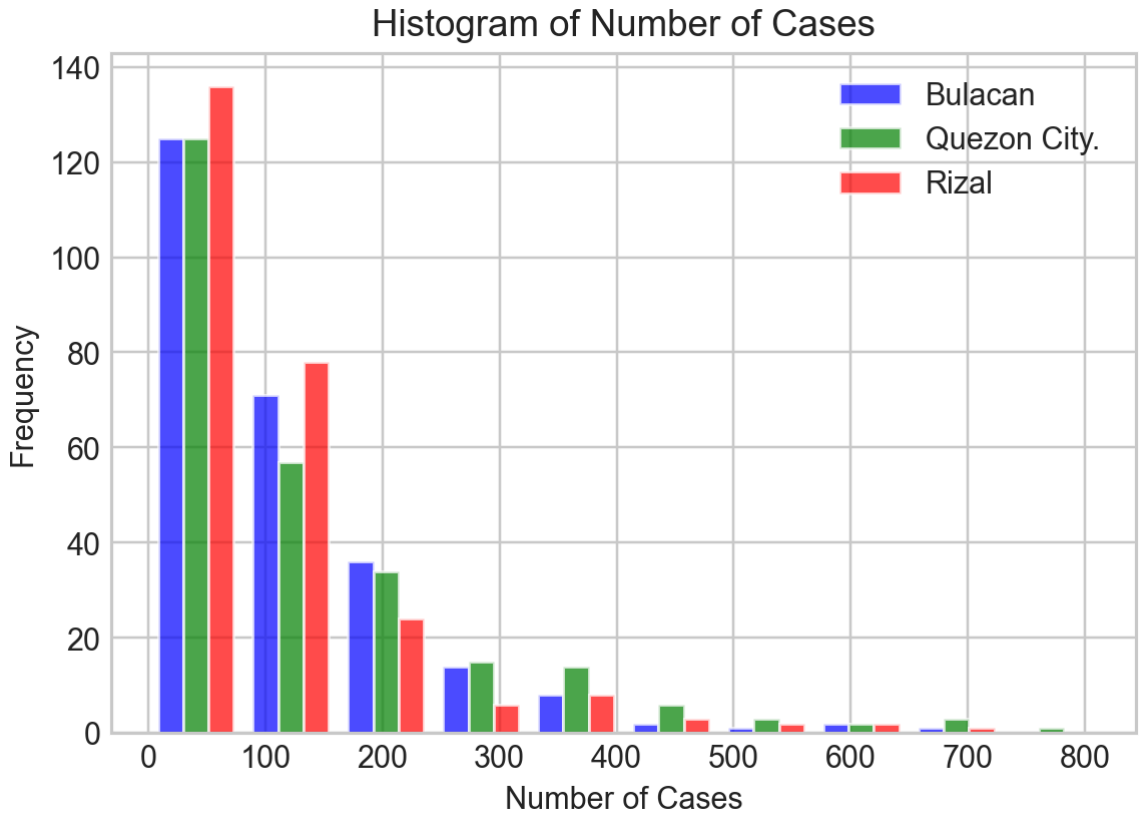
```
Number of Null Values in Bulacan df:
loc       0
region    0
cases     0
deaths    0
dtype: int64

Number of Null Values in Quezon df:
loc       0
region    0
cases     0
deaths    0
dtype: int64

Number of Null Values in Rizal df:
loc       0
region    0
cases     0
deaths    0
dtype: int64
```

In [462]:
```
# List of dataframes and corresponding labels
dfs = [df_dengue_bulacan, df_dengue_rizal, df_dengue_quezon]
labels = ['Bulacan', 'Rizal', 'Quezon']

# Compute descriptive statistics for each dataframe and combine them into one
df_describe = pd.concat([df.describe().assign(Location=label) for df, label in zip(dfs, labels)])

# Reset index and set 'Location' as the first index level
df_describe = df_describe.reset_index().set_index(['Location', 'index'])

# Pivot 'Location' level to the columns
df_describe = df_describe.unstack(level=0)

df_describe
```

Out[462]:

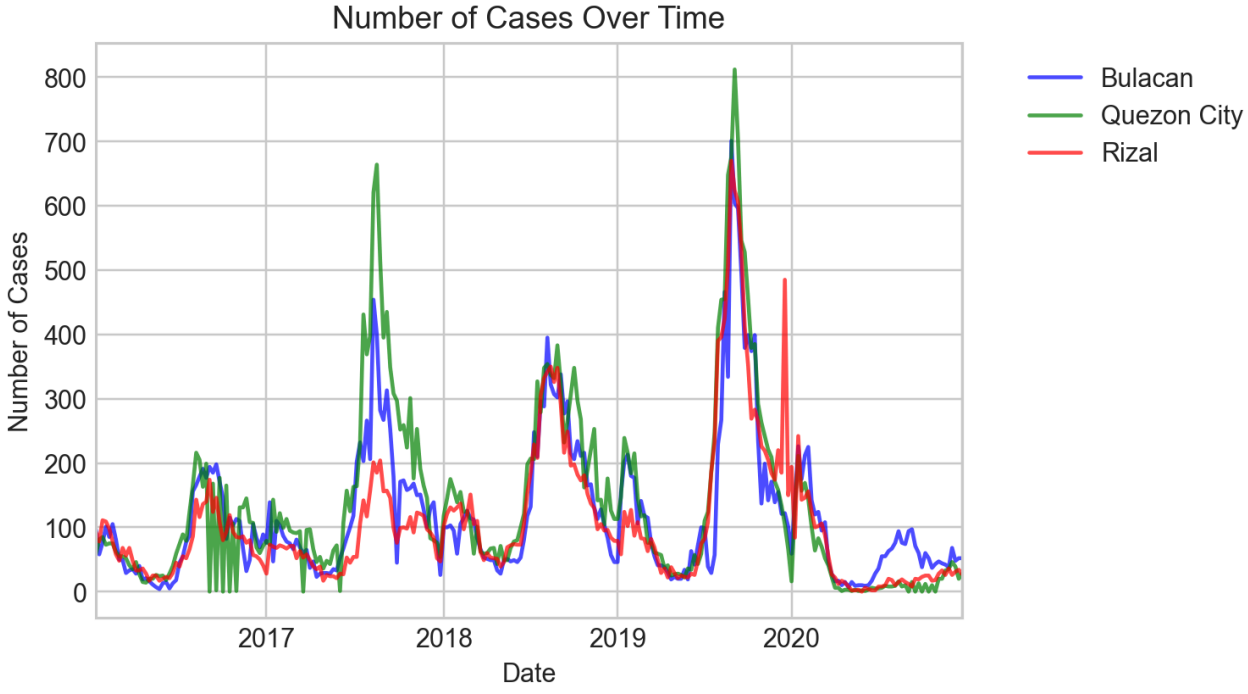
	cases			deaths		
Location	Bulacan	Quezon	Rizal	Bulacan	Quezon	Rizal
index						
25%	46.000000	36.750000	33.000000	0.000000	0.000000	0.000000
50%	84.000000	91.500000	77.000000	0.000000	1.000000	0.000000
75%	158.750000	178.750000	131.000000	1.000000	2.000000	1.000000
count	260.000000	260.000000	260.000000	260.000000	260.000000	260.000000
max	701.000000	812.000000	670.000000	4.000000	203.000000	7.000000
mean	118.588462	136.576923	106.742308	0.342308	4.650000	0.830769
min	4.000000	0.000000	0.000000	0.000000	0.000000	0.000000
std	110.414749	144.385310	110.052913	0.629195	23.357152	1.156739

In [463]:
```
plt.figure(figsize=(6, 4))
plt.hist([df_dengue_bulacan['cases'], df_dengue_quezon['cases'], df_dengue_rizal['cases']],
         bins=10, color=['blue', 'green', 'red'], alpha=0.7, label=['Bulacan', 'Quezon City. ', 'Rizal'])
plt.style.use('seaborn-whitegrid')
plt.title('Histogram of Number of Cases')
plt.xlabel('Number of Cases')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```



Binomial Distribution Plot

In [464]:
```
plt.figure(figsize=(6, 4))
plt.style.use('seaborn-whitegrid')
df_dengue_bulacan['cases'].plot(label='Bulacan', legend=True, color='blue', alpha=0.7)
df_dengue_quezon['cases'].plot(label='Quezon City', legend=True, color='green', alpha=0.7)
df_dengue_rizal['cases'].plot(label='Rizal', legend=True, color='red', alpha=0.7)

plt.title('Number of Cases Over Time')
plt.xlabel('Date')
plt.ylabel('Number of Cases')

plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

plt.show()
```

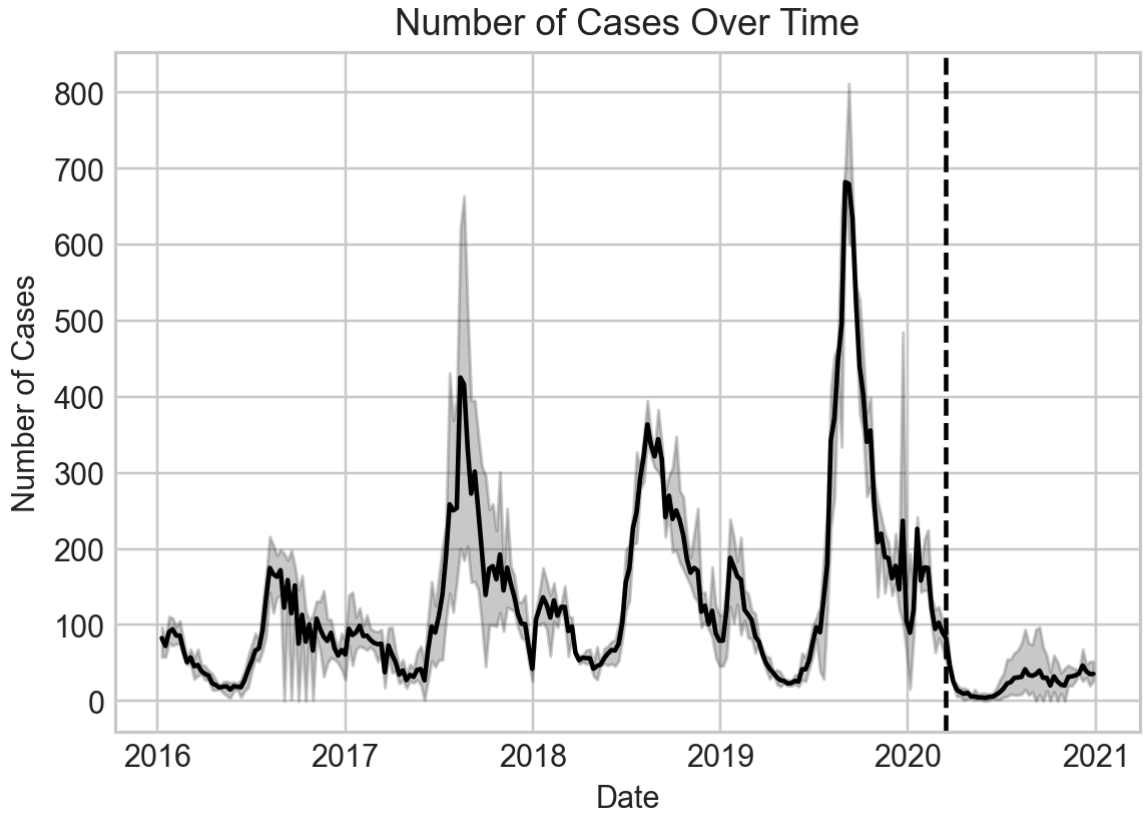


In [465]:
```
df_dengue.head()
```

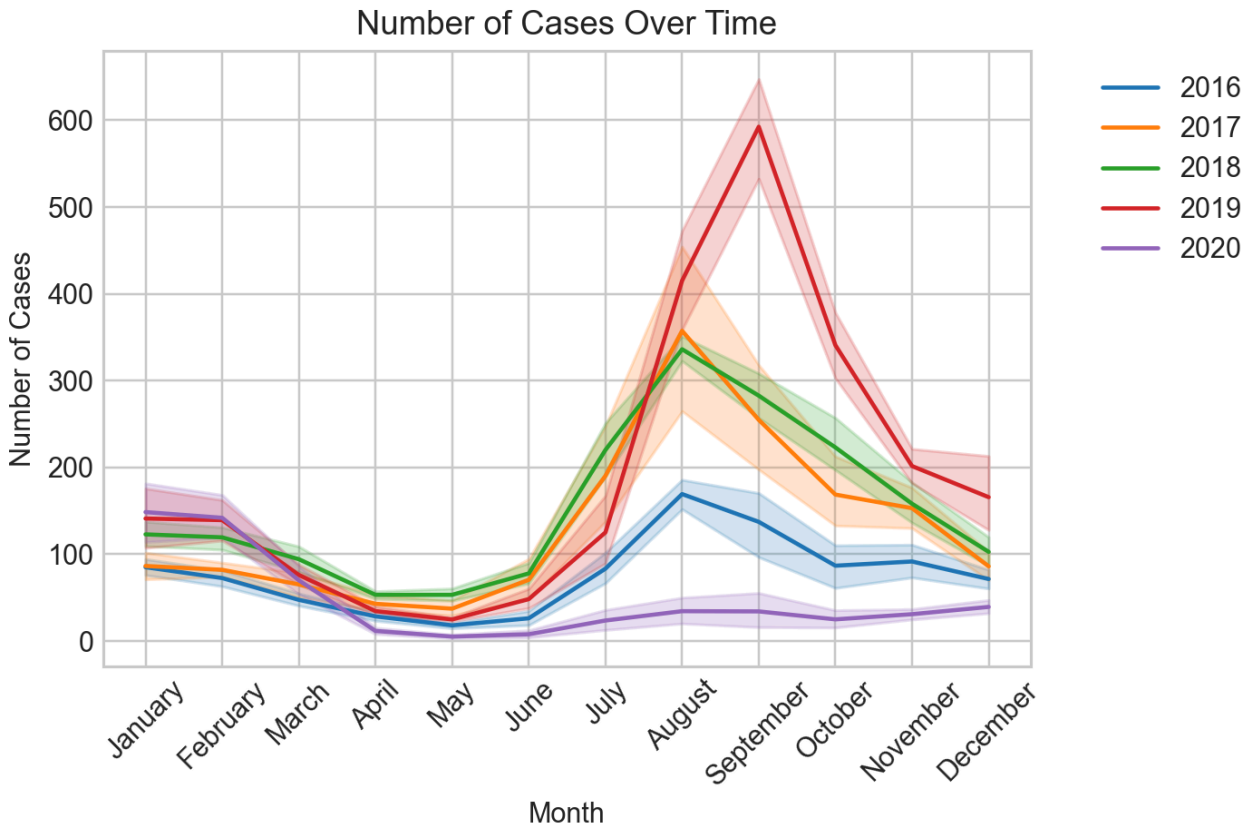
Out[465]:

	date	loc	region	cases	deaths
0	2016-01-10	BULACAN	REGION III-CENTRAL LUZON	97	0
52	2016-01-10	RIZAL	REGION IV-A-CALABARZON	93	0
104	2016-01-10	QUEZON CITY	NATIONAL CAPITAL REGION	58	0
105	2016-01-17	QUEZON CITY	NATIONAL CAPITAL REGION	81	0
1	2016-01-17	BULACAN	REGION III-CENTRAL LUZON	58	0

In [466]:
```
df_dengue = pd.concat([df_dengue_bulacan, df_dengue_rizal, df_dengue_quezon])

plt.figure(figsize=(6, 4))
plt.style.use('seaborn-whitegrid')
sns.lineplot(x='date', y='cases', color='black', data=df_dengue)

plt.title('Number of Cases Over Time')
plt.xlabel('Date')
plt.ylabel('Number of Cases')

plt.axvline(x=pd.to_datetime('2020-03-15'), color='black', linestyle='--')

plt.show()
```

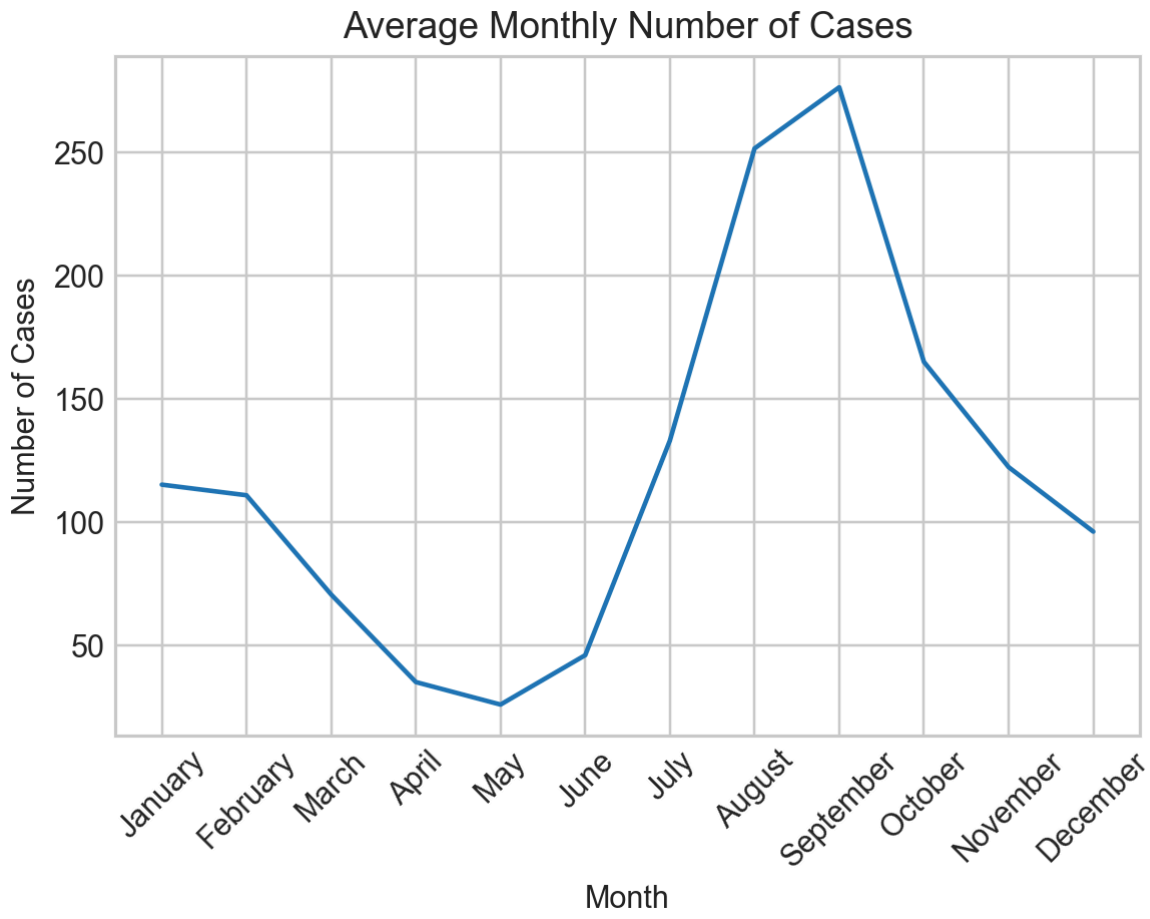


In [467]:
```
df_dengue['year'] = df_dengue.index.year
df_dengue['month'] = df_dengue.index.month_name()

plt.figure(figsize=(6, 4))
plt.style.use('seaborn-whitegrid')
sns.lineplot(x='month', y='cases', hue='year', data=df_dengue, palette='tab10')

plt.title('Number of Cases Over Time')
plt.xlabel('Month')
plt.ylabel('Number of Cases')
plt.xticks(rotation=45)

plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

plt.show()
```

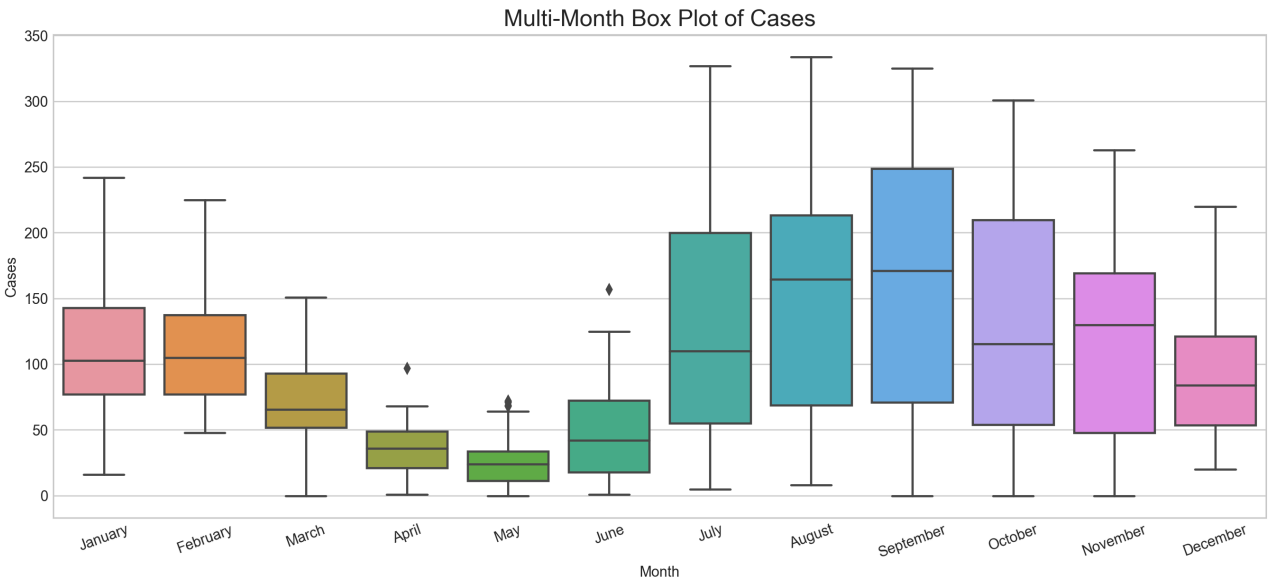


In [468]:
```
month_dengue = df_dengue.groupby(['month']).mean()
month_dengue.index = pd.to_datetime(month_dengue.index, format='%B')
month_dengue = month_dengue.sort_index()

plt.figure(figsize=(6, 4))
plt.style.use('seaborn-whitegrid')
sns.lineplot(x=month_dengue.index.month_name(), y='cases', data=month_dengue, palette='tab10')

plt.title('Average Monthly Number of Cases')
plt.xlabel('Month')
plt.ylabel('Number of Cases')
plt.xticks(rotation=45)

plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

plt.show()
```

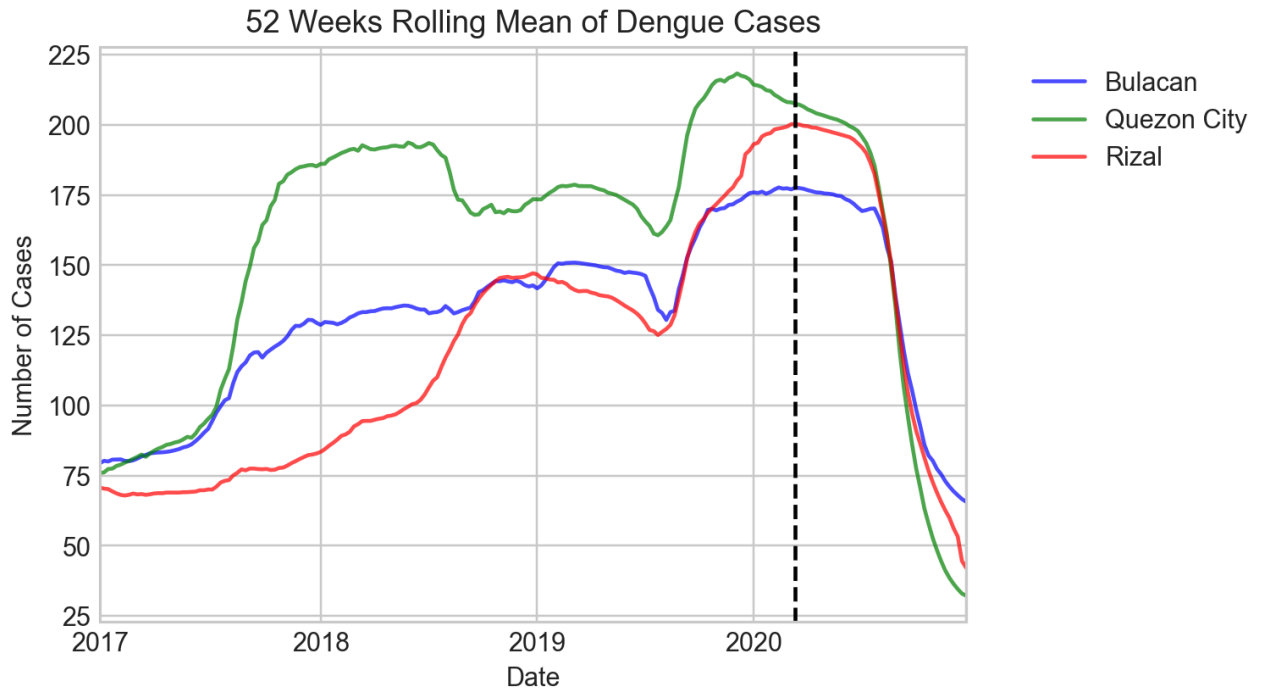
```
C:\Users\maryn\AppData\Local\Temp\ipykernel_27320\1563668013.py:1: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprecated. In a futur
e version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  month_dengue = df_dengue.groupby(['month']).mean()
No artists with labels found to put in legend.  Note that artists whose label start with an underscore are ignored when legend() is called with no argument.
```



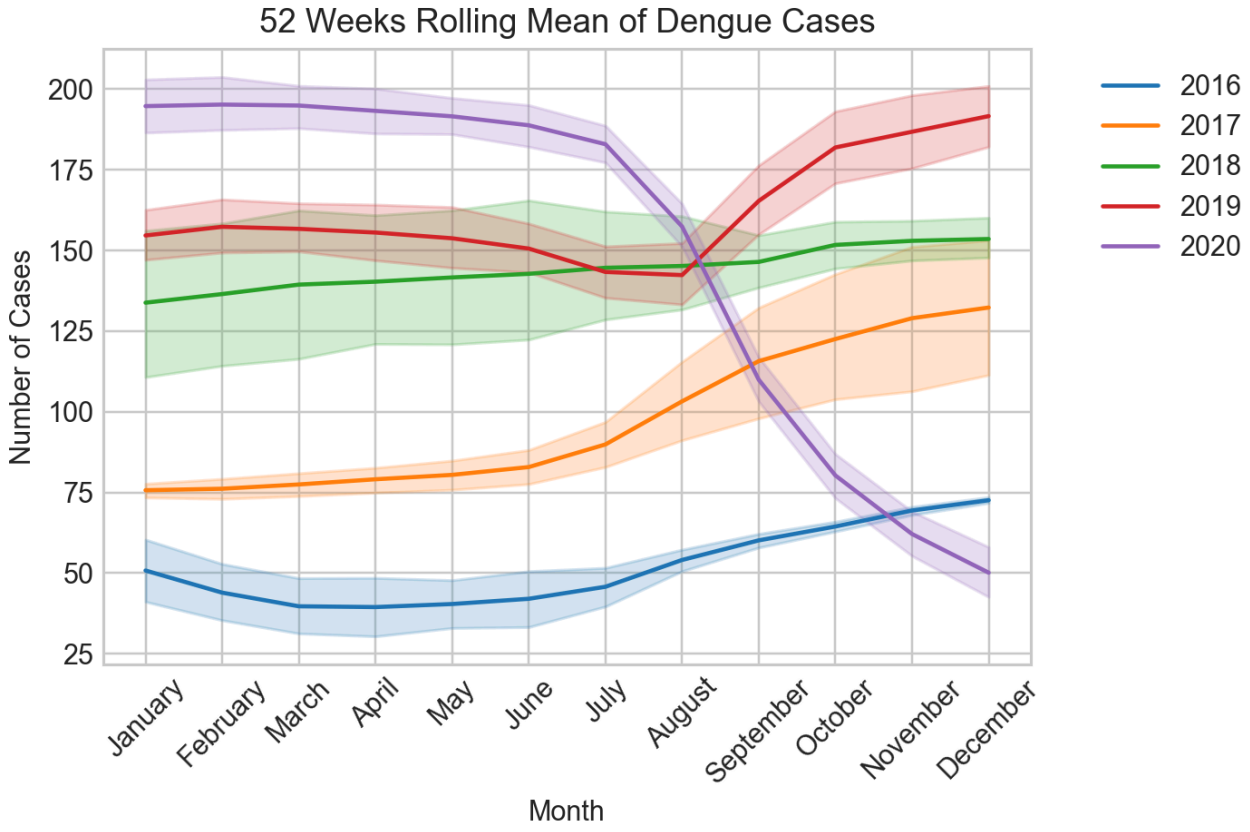
In [469]:
```
Q1 = df_dengue['cases'].quantile(0.25)
Q3 = df_dengue['cases'].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

df_dengue_no_outliers = df_dengue[(df_dengue['cases'] >= lower_bound) & (df_dengue['cases'] <= upper_bound)]

plt.figure(figsize=(15,6))
sns.boxplot(x='month', y='cases', data=df_dengue_no_outliers)

plt.xticks(range(12), calendar.month_name[1:13], rotation=20)
plt.title('Multi-Month Box Plot of Cases', fontsize=16)
plt.xlabel('Month')
plt.ylabel('Cases')
plt.show()
```



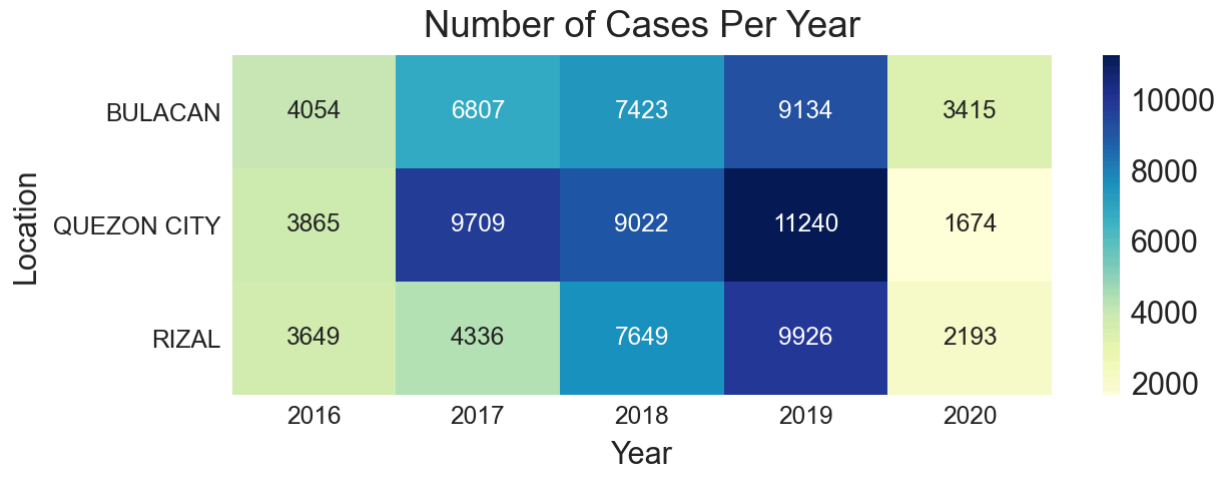
In [470]:
```
plt.figure(figsize=(6, 4))

mean_dengue_bulacan = df_dengue_bulacan['cases'].rolling(window=52).mean().dropna()
mean_dengue_quezon = df_dengue_quezon['cases'].rolling(window=52).mean().dropna()
mean_dengue_rizal = df_dengue_rizal['cases'].rolling(window=52).mean().dropna()

mean_dengue_bulacan.plot(label='Bulacan', legend=True, color='blue', alpha=0.7)
mean_dengue_quezon.plot(label='Quezon City', legend=True, color='green', alpha=0.7)
mean_dengue_rizal.plot(label='Rizal', legend=True, color='red', alpha=0.7)

plt.title("52 Weeks Rolling Mean of Dengue Cases")
plt.xlabel('Date')
plt.ylabel('Number of Cases')

plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

plt.axvline(x='2020-03-15', color='black', linestyle='--')

plt.show()
```



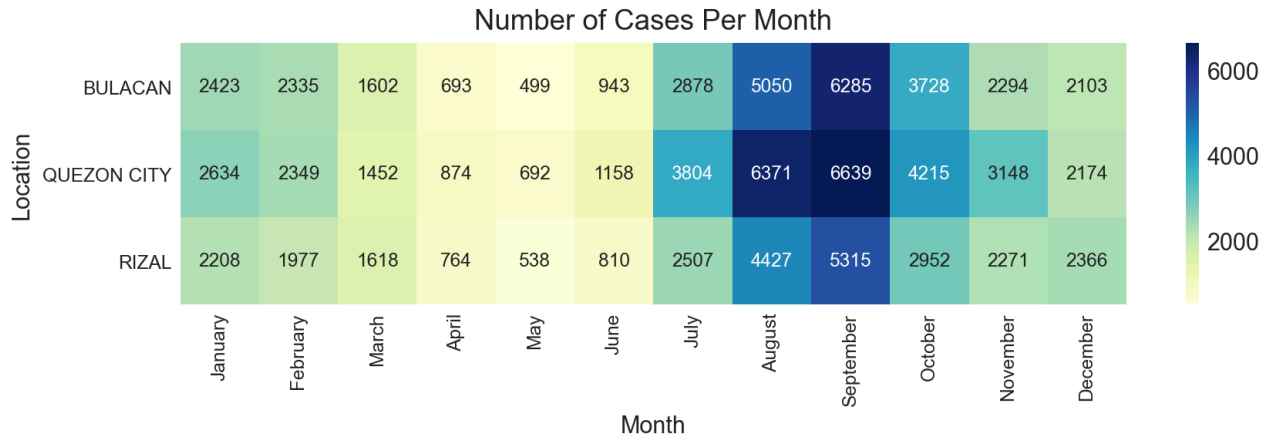
In [471]:
```
mean_dengue = df_dengue['cases'].rolling(window=52).mean().dropna()
mean_dengue = mean_dengue.to_frame()
mean_dengue['year'] = mean_dengue.index.year
mean_dengue['month'] = mean_dengue.index.month_name()

plt.figure(figsize=(6, 4))
plt.style.use('seaborn-whitegrid')
sns.lineplot(x='month', y='cases', hue='year', data=mean_dengue, palette='tab10')

plt.title('52 Weeks Rolling Mean of Dengue Cases')
plt.xlabel('Month')
plt.ylabel('Number of Cases')
plt.xticks(rotation=45)

plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

plt.show()
```



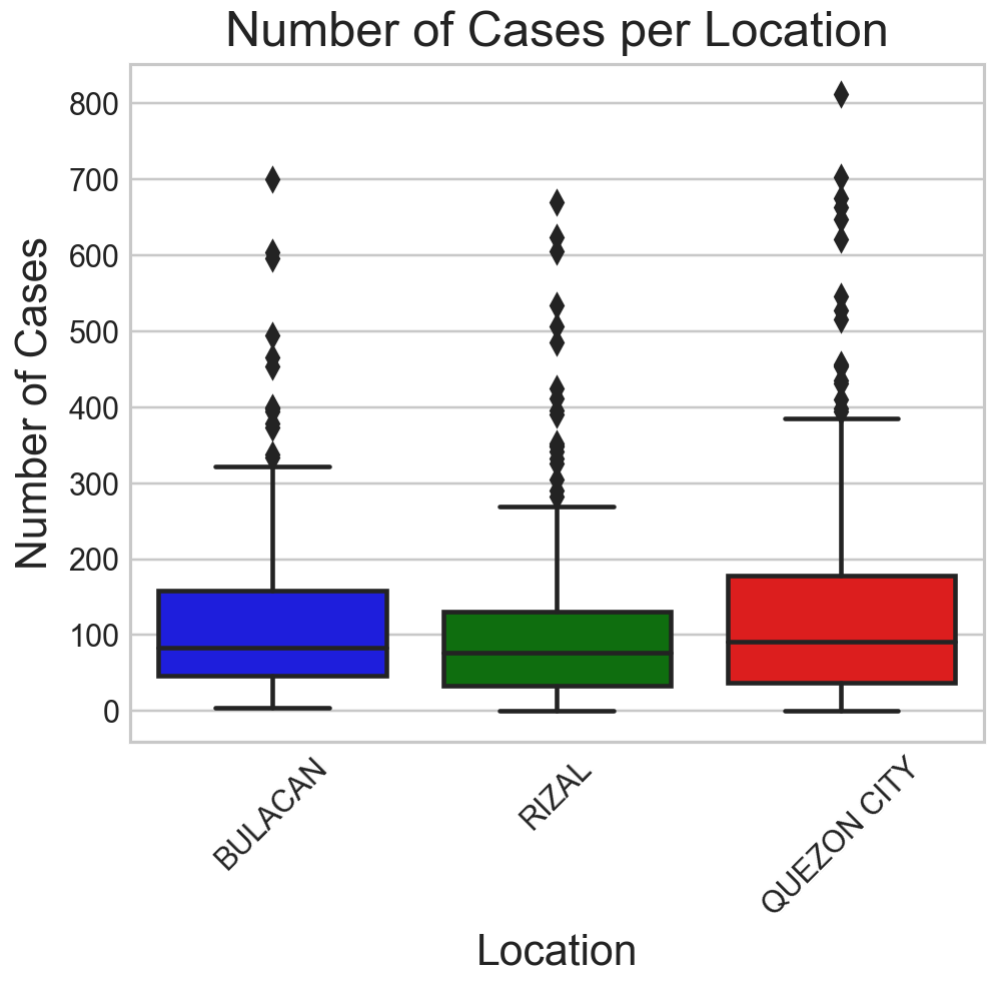
In [472]:
```
df_pivot = df_dengue.pivot_table(index='loc', columns='year', values='cases', aggfunc='sum')

plt.figure(figsize=(6, 2))
sns.heatmap(df_pivot, cmap='YlGnBu', annot=True, fmt=".0f", annot_kws={"size": 8})

plt.title('Number of Cases Per Year')
plt.xlabel('Year')
plt.ylabel('Location')
plt.tick_params(axis='both', which='major', labelsize=8)
plt.show()
```



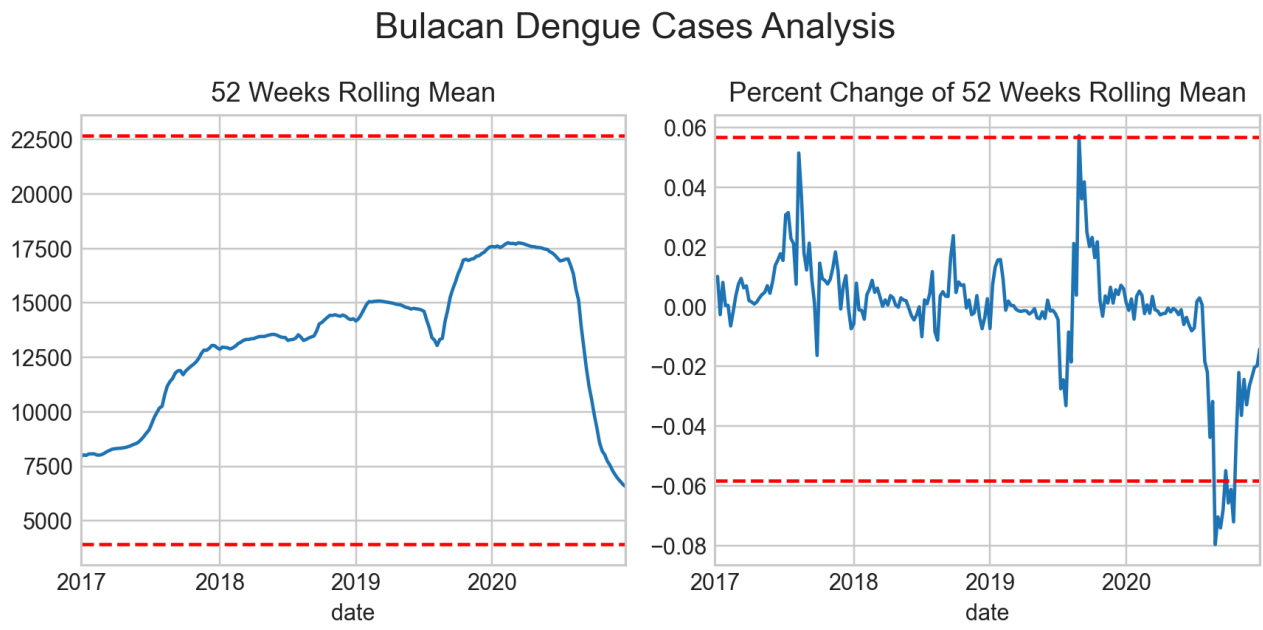
In [473]:
```
order = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December']
df_pivot = df_dengue.pivot_table(index='loc', columns='month', values='cases', aggfunc='sum')
df_pivot = df_pivot[order]

plt.figure(figsize=(9, 2))
sns.heatmap(df_pivot, cmap='YlGnBu', annot=True, fmt=".0f", annot_kws={"size": 8})

plt.title('Number of Cases Per Month')
plt.xlabel('Month')
plt.ylabel('Location')
plt.tick_params(axis='both', which='major', labelsize=8)
plt.show()
```



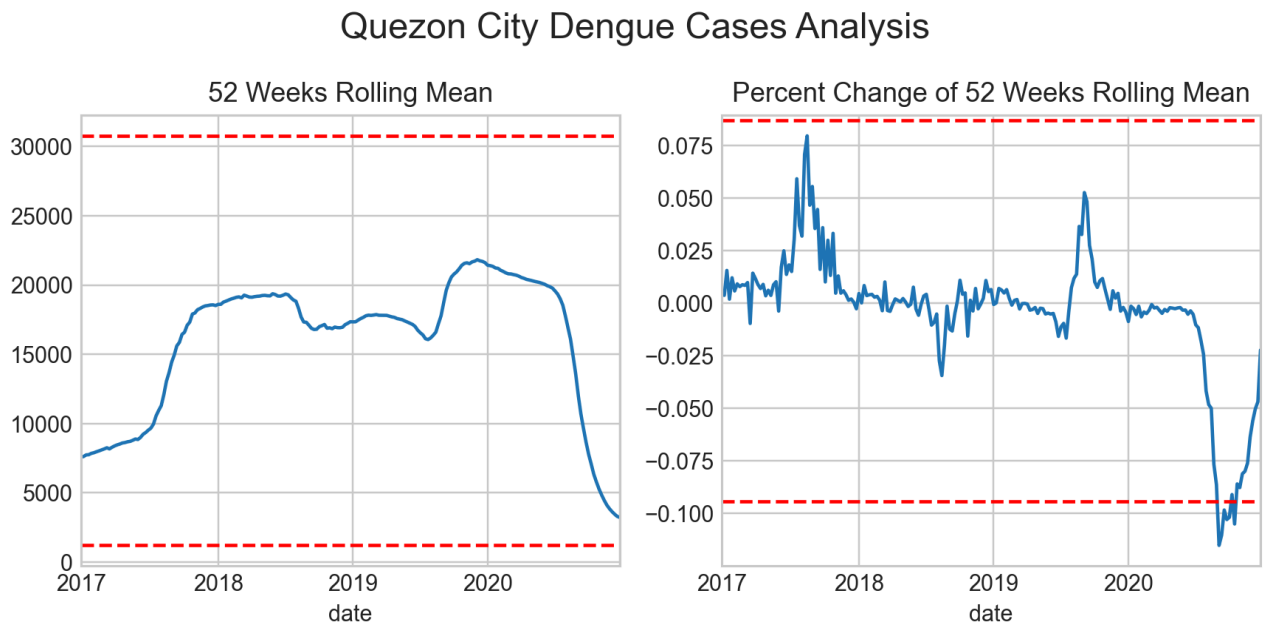
In [474]:
```
color_dict = {'BULACAN': 'blue', 'RIZAL': 'green', 'QUEZON CITY': 'red'}

plt.figure(figsize=(5, 4))
sns.set_style("whitegrid")
sns.boxplot(x='loc', y='cases', data=df_dengue, palette=color_dict)

plt.title('Number of Cases per Location', fontsize=16)
plt.xlabel('Location', fontsize=14)
plt.ylabel('Number of Cases', fontsize=14)
plt.xticks(rotation=45, fontsize=10)
plt.yticks(fontsize=10)

plt.show()
```



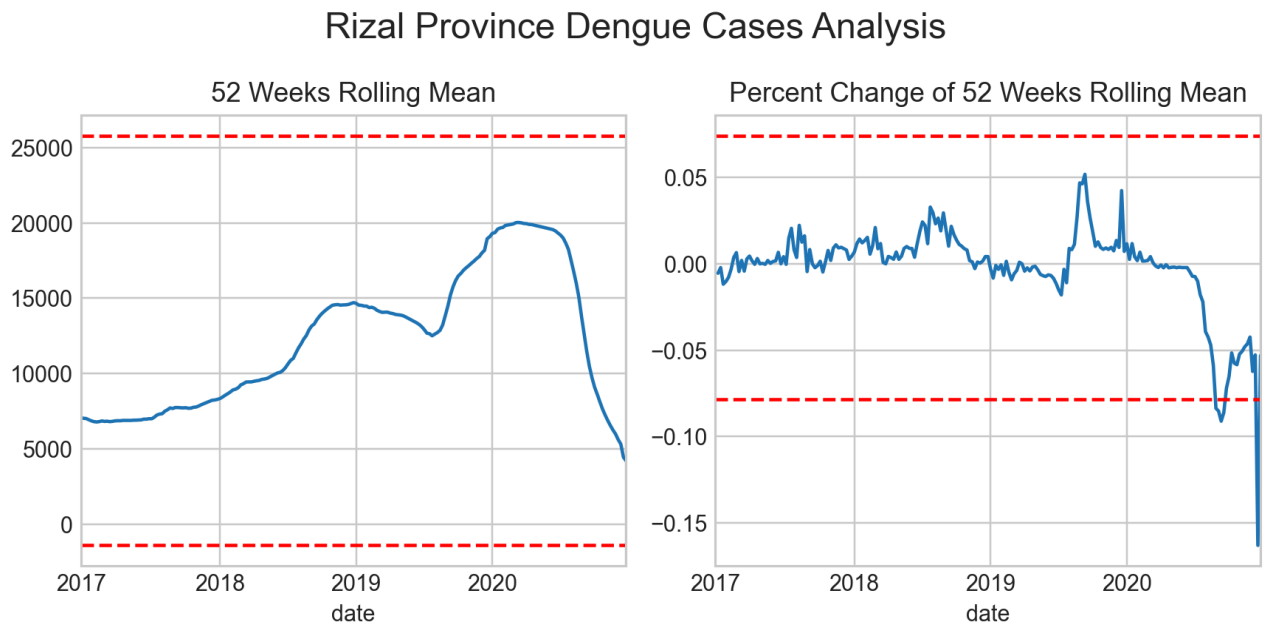
In [475]:
```
fig, axs = plt.subplots(1, 2, figsize=(8, 4))

df_loc = df_dengue_bulacan.copy()
rolling_mean  = df_loc['cases'].rolling(window=52).mean().mul(100).dropna()
pct_change_rolling_mean  = rolling_mean.to_frame()
pct_change_rolling_mean  = pct_change_rolling_mean['cases'].pct_change()

titles = ['52 Weeks Rolling Mean', 'Percent Change of 52 Weeks Rolling Mean']

for data, ax, title in zip([rolling_mean, pct_change_rolling_mean ], axs, titles):

    this_mean = data.mean()
    this_std = data.std()
    # Plot the data, with a window that is 3 standard deviations around the mean
    data.plot(ax=ax)
    ax.axhline(this_mean + this_std * 3, ls='--', c='r')
    ax.axhline(this_mean - this_std * 3, ls='--', c='r')
    ax.set_title(title)

fig.suptitle('Bulacan Dengue Cases Analysis', fontsize=16)

plt.tight_layout()
plt.show()
```



In [476]:
```
fig, axs = plt.subplots(1, 2, figsize=(8, 4))

df_loc = df_dengue_quezon.copy()
rolling_mean  = df_loc['cases'].rolling(window=52).mean().mul(100).dropna()
pct_change_rolling_mean  = rolling_mean.to_frame()
pct_change_rolling_mean  = pct_change_rolling_mean['cases'].pct_change()

titles = ['52 Weeks Rolling Mean', 'Percent Change of 52 Weeks Rolling Mean']

for data, ax, title in zip([rolling_mean, pct_change_rolling_mean ], axs, titles):
    # Calculate the mean / standard deviation for the data
    this_mean = data.mean()
    this_std = data.std()
    # Plot the data, with a window that is 3 standard deviations
    # around the mean
    data.plot(ax=ax)
    ax.axhline(this_mean + this_std * 3, ls='--', c='r')
    ax.axhline(this_mean - this_std * 3, ls='--', c='r')
    ax.set_title(title)

fig.suptitle('Quezon City Dengue Cases Analysis', fontsize=16)
plt.tight_layout()
plt.show()
```

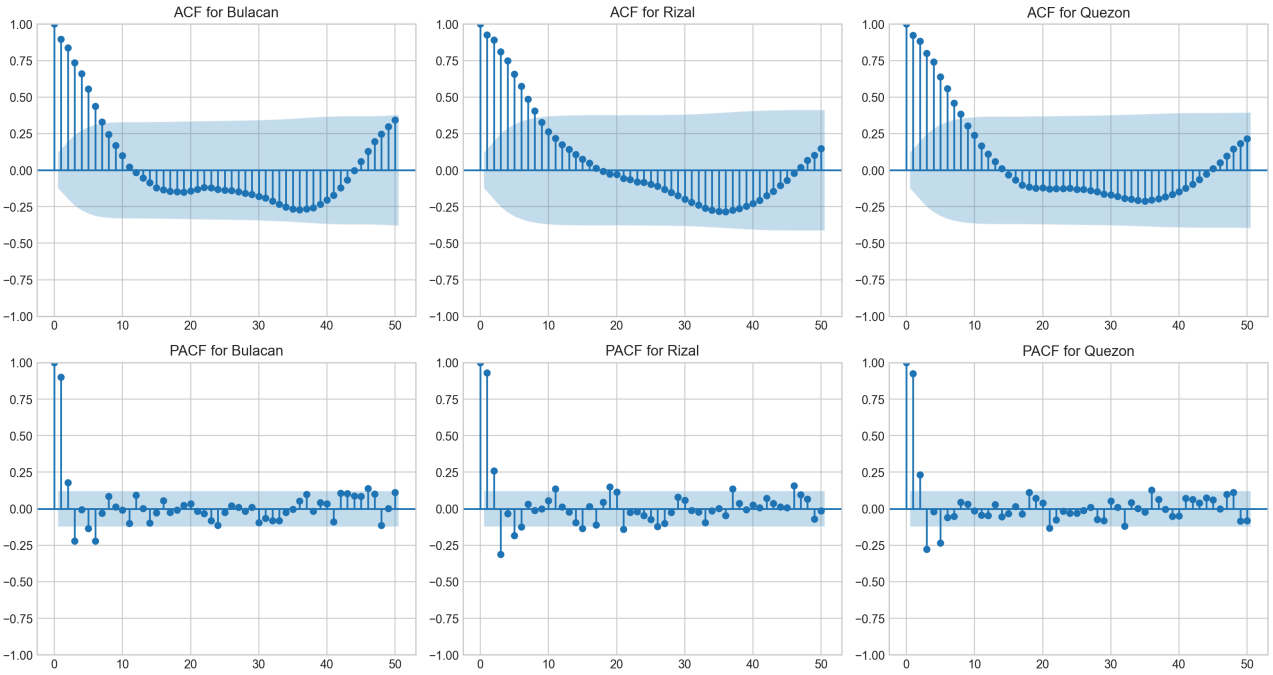


In [477]:
```
fig, axs = plt.subplots(1, 2, figsize=(8, 4))

df_loc = df_dengue_rizal.copy()
rolling_mean  = df_loc['cases'].rolling(window=52).mean().mul(100).dropna()
pct_change_rolling_mean  = rolling_mean.to_frame()
pct_change_rolling_mean  = pct_change_rolling_mean['cases'].pct_change()

titles = ['52 Weeks Rolling Mean', 'Percent Change of 52 Weeks Rolling Mean']

for data, ax, title in zip([rolling_mean, pct_change_rolling_mean ], axs, titles):
    # Calculate the mean / standard deviation for the data
    this_mean = data.mean()
    this_std = data.std()
    # Plot the data, with a window that is 3 standard deviations
    # around the mean
    data.plot(ax=ax)
    ax.axhline(this_mean + this_std * 3, ls='--', c='r')
    ax.axhline(this_mean - this_std * 3, ls='--', c='r')
    ax.set_title(title)

fig.suptitle('Rizal Province Dengue Cases Analysis', fontsize=16)
plt.tight_layout()
plt.show()
```



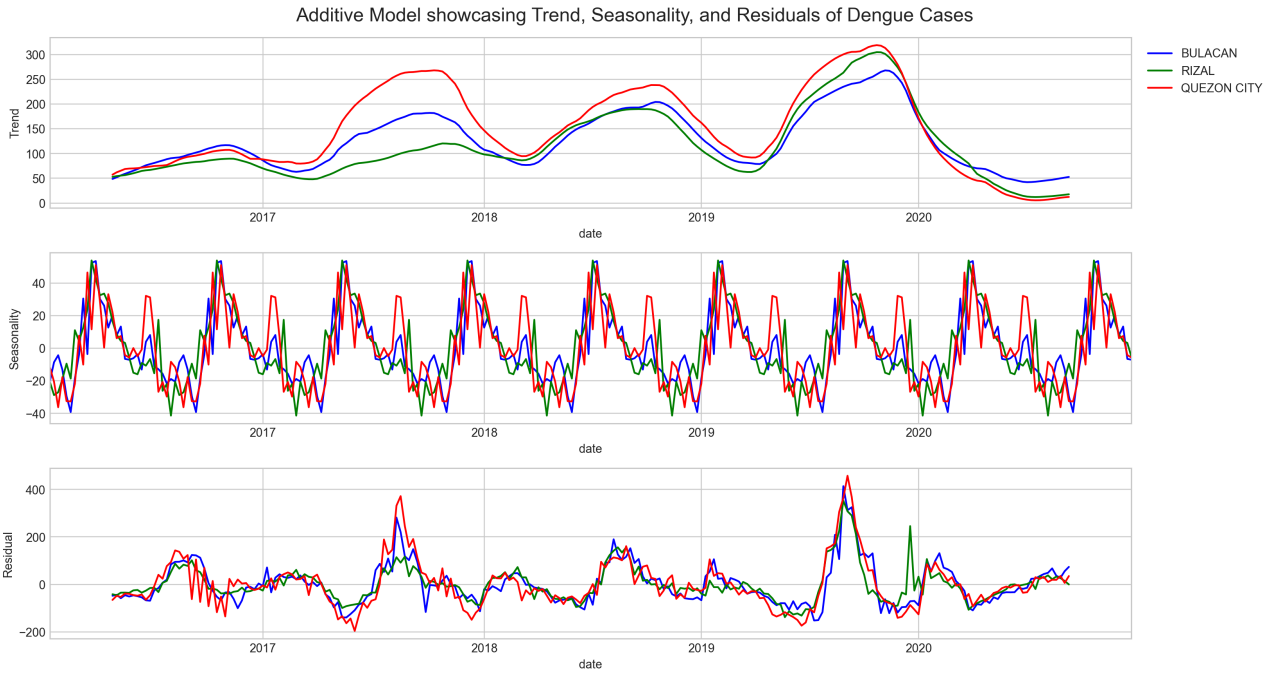
In [478]:
```
dfs = [df_dengue_bulacan, df_dengue_rizal, df_dengue_quezon]
labels = ['Bulacan', 'Rizal', 'Quezon']

fig, axs = plt.subplots(2, len(dfs), figsize=(15, 8))

for i, (df, label) in enumerate(zip(dfs, labels)):

    plot_acf(df['cases'], lags=50, ax=axs[0, i])
    plot_pacf(df['cases'], lags=50, ax=axs[1, i])

    axs[0, i].set_title(f'ACF for {label}')
    axs[1, i].set_title(f'PACF for {label}')

plt.tight_layout()
plt.show()
```

```
c:\Users\maryn\anaconda3\lib\site-packages\statsmodels\graphics\tsaplots.py:348: FutureWarning: The default method 'yw' can produce PACF values outside of the [-1,1]
interval. After 0.13, the default will change tounadjusted Yule-Walker ('ywm'). You can use this method now by setting method='ywm'.
  warnings.warn(
```



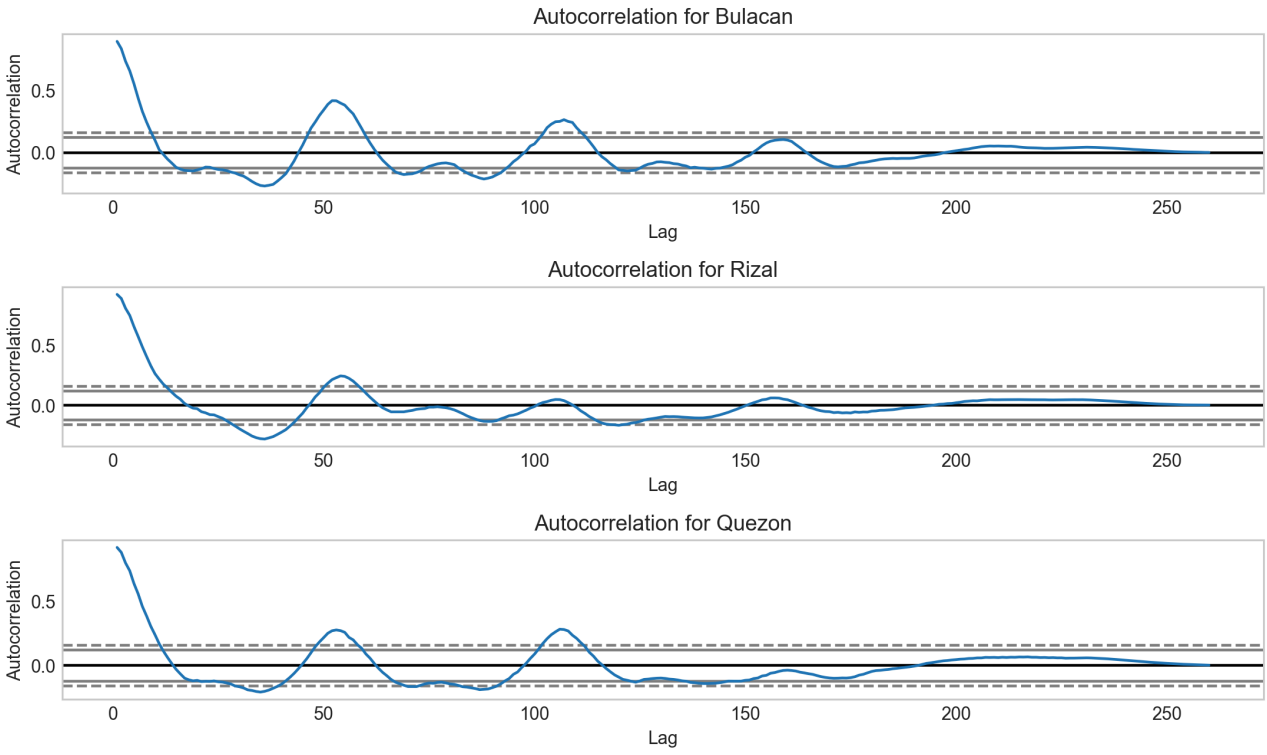
ACF: On this plot, there is a significant correlation at lag 1 that decreases after a few lags. This pattern indicates an autoregressive term. PACF: Significant correlations at the first few lags, especially at lag 1 and 2, followed by not so significant correlations. There is an autoregressive term in the data, supporting what was seen in the ACF Plot.

In [479]:
```
dfs = [df_dengue_bulacan, df_dengue_rizal, df_dengue_quezon]
labels = ['BULACAN', 'RIZAL', 'QUEZON CITY']

color_dict = {'BULACAN': 'blue', 'RIZAL': 'green', 'QUEZON CITY': 'red'}

fig, axs = plt.subplots(3, 1, figsize=(15, 8))

for df, label in zip(dfs, labels):
    res = seasonal_decompose(df['cases'], model="additive", period=30)

    color = color_dict.get(label)

    res.trend.plot(ax=axs[0], label=label, color=color)
    res.seasonal.plot(ax=axs[1], label=label, color=color)
    res.resid.plot(ax=axs[2], label=label, color=color)

axs[0].set_ylabel('Trend')
axs[1].set_ylabel('Seasonality')
axs[2].set_ylabel('Residual')
axs[0].legend(bbox_to_anchor=(1.005, 1), loc='upper left')

plt.suptitle('Additive Model showcasing Trend, Seasonality, and Residuals of Dengue Cases', fontsize=16)
plt.tight_layout()
plt.show()
```



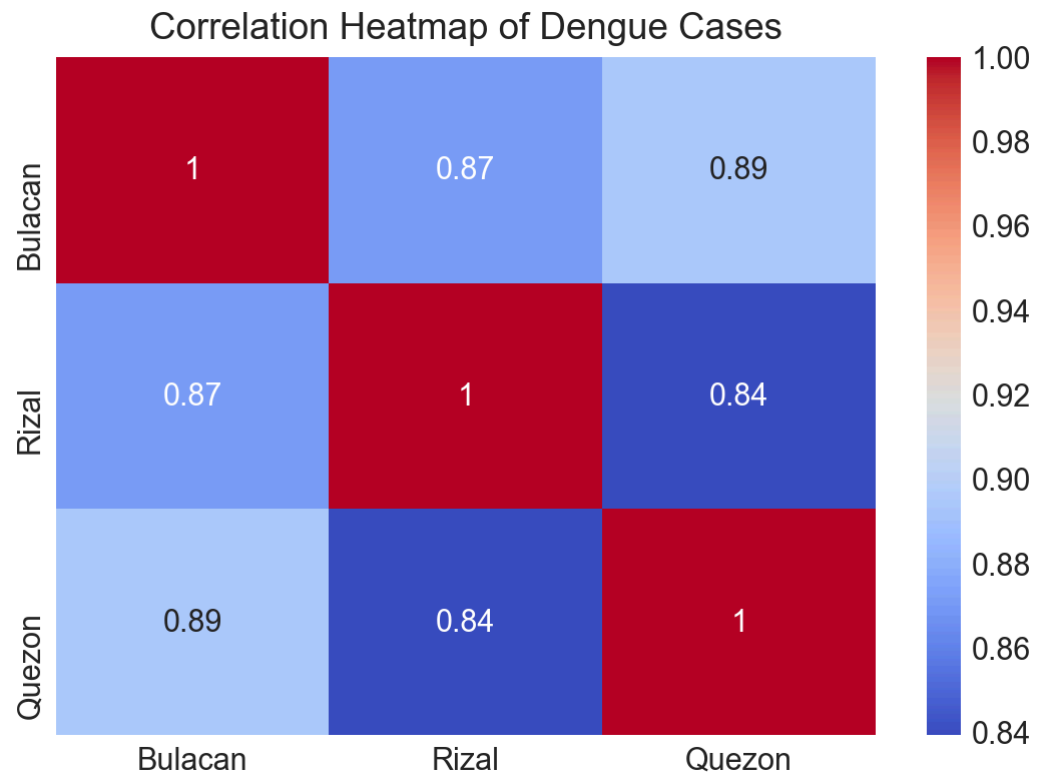
In [480]:

```
dfs = [df_dengue_bulacan, df_dengue_rizal, df_dengue_quezon]
labels = ['Bulacan', 'Rizal', 'Quezon']

fig, axs = plt.subplots(len(dfs), figsize=(10, 2*len(dfs)))

for df, label, ax in zip(dfs, labels, axs):
    autocorrelation_plot(df['cases'].tolist(), ax=ax)
    ax.set_title(f'Autocorrelation for {label}')

plt.tight_layout()
plt.show()
```



In [481]:

```
dfs = [df_dengue_bulacan['cases'], df_dengue_rizal['cases'], df_dengue_quezon['cases']]
labels = ['Bulacan', 'Rizal', 'Quezon']
df_all = pd.concat([df.rename(label) for df, label in zip(dfs, labels)], axis=1)
corr = df_all.corr()

plt.figure(figsize=(6, 4))
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap of Dengue Cases')
plt.show()
```



df_climate

In [482]:
```
df_climate.head()
```

Out[482]:

	loc	date	MAX__AIRMASS	MIN__AIRMASS	MEAN__AIRMASS	SUM__AIRMASS	STD__AIRMASS	MAX__ALLSKY_KT	MIN__ALLSKY_KT	MEAN__ALLSKY_KT	...	MAX__WS50M_RANGE	MIN__WS50M
0	BULACAN	2016-01-10	-999.0	-999.0	-999.0	-6993.0	0.0	0.71	0.60	0.671429	...	3.34	
1	BULACAN	2016-01-17	-999.0	-999.0	-999.0	-6993.0	0.0	0.69	0.61	0.661429	...	3.16	
2	BULACAN	2016-01-24	-999.0	-999.0	-999.0	-6993.0	0.0	0.66	0.47	0.612857	...	5.77	
3	BULACAN	2016-01-31	-999.0	-999.0	-999.0	-6993.0	0.0	0.63	0.38	0.541429	...	6.16	
4	BULACAN	2016-02-07	-999.0	-999.0	-999.0	-6993.0	0.0	0.68	0.20	0.495714	...	7.05	

5 rows × 502 columns

In [483]:
```
print(f"Columns: \n{np.array(df_climate.columns)}")
```

Columns:

```
['loc' 'date' 'MAX__AIRMASS' 'MIN__AIRMASS' 'MEAN__AIRMASS' 'SUM__AIRMASS'
 'STD__AIRMASS' 'MAX__ALLSKY_KT' 'MIN__ALLSKY_KT' 'MEAN__ALLSKY_KT'
 'SUM__ALLSKY_KT' 'STD__ALLSKY_KT' 'MAX__ALLSKY_SFC_LW_DWN'
 'MIN__ALLSKY_SFC_LW_DWN' 'MEAN__ALLSKY_SFC_LW_DWN'
 'SUM__ALLSKY_SFC_LW_DWN' 'STD__ALLSKY_SFC_LW_DWN' 'MAX__ALLSKY_SFC_LW_UP'
 'MIN__ALLSKY_SFC_LW_UP' 'MEAN__ALLSKY_SFC_LW_UP' 'SUM__ALLSKY_SFC_LW_UP'
 'STD__ALLSKY_SFC_LW_UP' 'MAX__ALLSKY_SFC_PAR_TOT'
 'MIN__ALLSKY_SFC_PAR_TOT' 'MEAN__ALLSKY_SFC_PAR_TOT'
 'SUM__ALLSKY_SFC_PAR_TOT' 'STD__ALLSKY_SFC_PAR_TOT'
 'MAX__ALLSKY_SFC_SW_DIFF' 'MIN__ALLSKY_SFC_SW_DIFF'
 'MEAN__ALLSKY_SFC_SW_DIFF' 'SUM__ALLSKY_SFC_SW_DIFF'
 'STD__ALLSKY_SFC_SW_DIFF' 'MAX__ALLSKY_SFC_SW_DNI'
 'MIN__ALLSKY_SFC_SW_DNI' 'MEAN__ALLSKY_SFC_SW_DNI'
 'SUM__ALLSKY_SFC_SW_DNI' 'STD__ALLSKY_SFC_SW_DNI'
 'MAX__ALLSKY_SFC_SW_DWN' 'MIN__ALLSKY_SFC_SW_DWN'
 'MEAN__ALLSKY_SFC_SW_DWN' 'SUM__ALLSKY_SFC_SW_DWN'
 'STD__ALLSKY_SFC_SW_DWN' 'MAX__ALLSKY_SFC_SW_UP' 'MIN__ALLSKY_SFC_SW_UP'
 'MEAN__ALLSKY_SFC_SW_UP' 'SUM__ALLSKY_SFC_SW_UP' 'STD__ALLSKY_SFC_SW_UP'
 'MAX__ALLSKY_SFC_UVA' 'MIN__ALLSKY_SFC_UVA' 'MEAN__ALLSKY_SFC_UVA'
 'SUM__ALLSKY_SFC_UVA' 'STD__ALLSKY_SFC_UVA' 'MAX__ALLSKY_SFC_UVB'
 'MIN__ALLSKY_SFC_UVB' 'MEAN__ALLSKY_SFC_UVB' 'SUM__ALLSKY_SFC_UVB'
 'STD__ALLSKY_SFC_UVB' 'MAX__ALLSKY_SFC_UV_INDEX'
 'MIN__ALLSKY_SFC_UV_INDEX' 'MEAN__ALLSKY_SFC_UV_INDEX'
 'SUM__ALLSKY_SFC_UV_INDEX' 'STD__ALLSKY_SFC_UV_INDEX'
 'MAX__ALLSKY_SRF_ALB' 'MIN__ALLSKY_SRF_ALB' 'MEAN__ALLSKY_SRF_ALB'
 'SUM__ALLSKY_SRF_ALB' 'STD__ALLSKY_SRF_ALB' 'MAX__AOD_55' 'MIN__AOD_55'
 'MEAN__AOD_55' 'SUM__AOD_55' 'STD__AOD_55' 'MAX__AOD_84' 'MIN__AOD_84'
 'MEAN__AOD_84' 'SUM__AOD_84' 'STD__AOD_84' 'MAX__CDD0' 'MIN__CDD0'
 'MEAN__CDD0' 'SUM__CDD0' 'STD__CDD0' 'MAX__CDD10' 'MIN__CDD10'
 'MEAN__CDD10' 'SUM__CDD10' 'STD__CDD10' 'MAX__CDD18_3' 'MIN__CDD18_3'
 'MEAN__CDD18_3' 'SUM__CDD18_3' 'STD__CDD18_3' 'MAX__CLOUD_AMT'
 'MIN__CLOUD_AMT' 'MEAN__CLOUD_AMT' 'SUM__CLOUD_AMT' 'STD__CLOUD_AMT'
 'MAX__CLOUD_AMT_DAY' 'MIN__CLOUD_AMT_DAY' 'MEAN__CLOUD_AMT_DAY'
 'SUM__CLOUD_AMT_DAY' 'STD__CLOUD_AMT_DAY' 'MAX__CLOUD_AMT_NIGHT'
 'MIN__CLOUD_AMT_NIGHT' 'MEAN__CLOUD_AMT_NIGHT' 'SUM__CLOUD_AMT_NIGHT'
 'STD__CLOUD_AMT_NIGHT' 'MAX__CLOUD_OD' 'MIN__CLOUD_OD' 'MEAN__CLOUD_OD'
 'SUM__CLOUD_OD' 'STD__CLOUD_OD' 'MAX__CLRSKY_DAYS' 'MIN__CLRSKY_DAYS'
 'MEAN__CLRSKY_DAYS' 'SUM__CLRSKY_DAYS' 'STD__CLRSKY_DAYS'
 'MAX__CLRSKY_KT' 'MIN__CLRSKY_KT' 'MEAN__CLRSKY_KT' 'SUM__CLRSKY_KT'
 'STD__CLRSKY_KT' 'MAX__CLRSKY_SFC_LW_DWN' 'MIN__CLRSKY_SFC_LW_DWN'
 'MEAN__CLRSKY_SFC_LW_DWN' 'SUM__CLRSKY_SFC_LW_DWN'
 'STD__CLRSKY_SFC_LW_DWN' 'MAX__CLRSKY_SFC_LW_UP' 'MIN__CLRSKY_SFC_LW_UP'
 'MEAN__CLRSKY_SFC_LW_UP' 'SUM__CLRSKY_SFC_LW_UP' 'STD__CLRSKY_SFC_LW_UP'
 'MAX__CLRSKY_SFC_PAR_TOT' 'MIN__CLRSKY_SFC_PAR_TOT'
 'MEAN__CLRSKY_SFC_PAR_TOT' 'SUM__CLRSKY_SFC_PAR_TOT'
 'STD__CLRSKY_SFC_PAR_TOT' 'MAX__CLRSKY_SFC_SW_DIFF'
 'MIN__CLRSKY_SFC_SW_DIFF' 'MEAN__CLRSKY_SFC_SW_DIFF'
 'SUM__CLRSKY_SFC_SW_DIFF' 'STD__CLRSKY_SFC_SW_DIFF'
 'MAX__CLRSKY_SFC_SW_DNI' 'MIN__CLRSKY_SFC_SW_DNI'
 'MEAN__CLRSKY_SFC_SW_DNI' 'SUM__CLRSKY_SFC_SW_DNI'
 'STD__CLRSKY_SFC_SW_DNI' 'MAX__CLRSKY_SFC_SW_DWN'
 'MIN__CLRSKY_SFC_SW_DWN' 'MEAN__CLRSKY_SFC_SW_DWN'
 'SUM__CLRSKY_SFC_SW_DWN' 'STD__CLRSKY_SFC_SW_DWN' 'MAX__CLRSKY_SFC_SW_UP'
 'MIN__CLRSKY_SFC_SW_UP' 'MEAN__CLRSKY_SFC_SW_UP' 'SUM__CLRSKY_SFC_SW_UP'
 'STD__CLRSKY_SFC_SW_UP' 'MAX__CLRSKY_SRF_ALB' 'MIN__CLRSKY_SRF_ALB'
 'MEAN__CLRSKY_SRF_ALB' 'SUM__CLRSKY_SRF_ALB' 'STD__CLRSKY_SRF_ALB'
 'MAX__DISPH' 'MIN__DISPH' 'MEAN__DISPH' 'SUM__DISPH' 'STD__DISPH'
 'MAX__EVLAND' 'MIN__EVLAND' 'MEAN__EVLAND' 'SUM__EVLAND' 'STD__EVLAND'
 'MAX__EVPTRNS' 'MIN__EVPTRNS' 'MEAN__EVPTRNS' 'SUM__EVPTRNS'
 'STD__EVPTRNS' 'MAX__FROST_DAYS' 'MIN__FROST_DAYS' 'MEAN__FROST_DAYS'
 'SUM__FROST_DAYS' 'STD__FROST_DAYS' 'MAX__FRSEAICE' 'MIN__FRSEAICE'
 'MEAN__FRSEAICE' 'SUM__FRSEAICE' 'STD__FRSEAICE' 'MAX__FRSNO'
 'MIN__FRSNO' 'MEAN__FRSNO' 'SUM__FRSNO' 'STD__FRSNO' 'MAX__GWETPROF'
 'MIN__GWETPROF' 'MEAN__GWETPROF' 'SUM__GWETPROF' 'STD__GWETPROF'
 'MAX__GWETROOT' 'MIN__GWETROOT' 'MEAN__GWETROOT' 'SUM__GWETROOT'
 'STD__GWETROOT' 'MAX__GWETTOP' 'MIN__GWETTOP' 'MEAN__GWETTOP'
 'SUM__GWETTOP' 'STD__GWETTOP' 'MAX__HDD0' 'MIN__HDD0' 'MEAN__HDD0'
 'SUM__HDD0' 'STD__HDD0' 'MAX__HDD10' 'MIN__HDD10' 'MEAN__HDD10'
 'SUM__HDD10' 'STD__HDD10' 'MAX__HDD18_3' 'MIN__HDD18_3' 'MEAN__HDD18_3'
 'SUM__HDD18_3' 'STD__HDD18_3' 'MAX__MIDDAY_INSOL' 'MIN__MIDDAY_INSOL'
 'MEAN__MIDDAY_INSOL' 'SUM__MIDDAY_INSOL' 'STD__MIDDAY_INSOL'
 'MAX__PBLTOP' 'MIN__PBLTOP' 'MEAN__PBLTOP' 'SUM__PBLTOP' 'STD__PBLTOP'
 'MAX__PRECSNOLAND' 'MIN__PRECSNOLAND' 'MEAN__PRECSNOLAND'
 'SUM__PRECSNOLAND' 'STD__PRECSNOLAND' 'MAX__PRECTOTCORR'
 'MIN__PRECTOTCORR' 'MEAN__PRECTOTCORR' 'SUM__PRECTOTCORR'
 'STD__PRECTOTCORR' 'MAX__PS' 'MIN__PS' 'MEAN__PS' 'SUM__PS' 'STD__PS'
 'MAX__PW' 'MIN__PW' 'MEAN__PW' 'SUM__PW' 'STD__PW' 'MAX__QV10M'
 'MIN__QV10M' 'MEAN__QV10M' 'SUM__QV10M' 'STD__QV10M' 'MAX__QV2M'
 'MIN__QV2M' 'MEAN__QV2M' 'SUM__QV2M' 'STD__QV2M' 'MAX__RH2M' 'MIN__RH2M'
 'MEAN__RH2M' 'SUM__RH2M' 'STD__RH2M' 'MAX__RHOA' 'MIN__RHOA' 'MEAN__RHOA'
 'SUM__RHOA' 'STD__RHOA' 'MAX__SG_NOON' 'MIN__SG_NOON' 'MEAN__SG_NOON'
 'SUM__SG_NOON' 'STD__SG_NOON' 'MAX__SLP' 'MIN__SLP' 'MEAN__SLP'
 'SUM__SLP' 'STD__SLP' 'MAX__SNODP' 'MIN__SNODP' 'MEAN__SNODP'
 'SUM__SNODP' 'STD__SNODP' 'MAX__T10M' 'MIN__T10M' 'MEAN__T10M'
 'SUM__T10M' 'STD__T10M' 'MAX__T10M_MAX' 'MIN__T10M_MAX' 'MEAN__T10M_MAX'
 'SUM__T10M_MAX' 'STD__T10M_MAX' 'MAX__T10M_MIN' 'MIN__T10M_MIN'
 'MEAN__T10M_MIN' 'SUM__T10M_MIN' 'STD__T10M_MIN' 'MAX__T10M_RANGE'
 'MIN__T10M_RANGE' 'MEAN__T10M_RANGE' 'SUM__T10M_RANGE' 'STD__T10M_RANGE'
 'MAX__T2M' 'MIN__T2M' 'MEAN__T2M' 'SUM__T2M' 'STD__T2M' 'MAX__T2MDEW'
 'MIN__T2MDEW' 'MEAN__T2MDEW' 'SUM__T2MDEW' 'STD__T2MDEW' 'MAX__T2MWET'
 'MIN__T2MWET' 'MEAN__T2MWET' 'SUM__T2MWET' 'STD__T2MWET' 'MAX__T2M_MAX'
 'MIN__T2M_MAX' 'MEAN__T2M_MAX' 'SUM__T2M_MAX' 'STD__T2M_MAX'
 'MAX__T2M_MIN' 'MIN__T2M_MIN' 'MEAN__T2M_MIN' 'SUM__T2M_MIN'
 'STD__T2M_MIN' 'MAX__T2M_RANGE' 'MIN__T2M_RANGE' 'MEAN__T2M_RANGE'
 'SUM__T2M_RANGE' 'STD__T2M_RANGE' 'MAX__TO3' 'MIN__TO3' 'MEAN__TO3'
 'SUM__TO3' 'STD__TO3' 'MAX__TOA_SW_DNI' 'MIN__TOA_SW_DNI'
 'MEAN__TOA_SW_DNI' 'SUM__TOA_SW_DNI' 'STD__TOA_SW_DNI' 'MAX__TOA_SW_DWN'
 'MIN__TOA_SW_DWN' 'MEAN__TOA_SW_DWN' 'SUM__TOA_SW_DWN' 'STD__TOA_SW_DWN'
 'MAX__TQV' 'MIN__TQV' 'MEAN__TQV' 'SUM__TQV' 'STD__TQV' 'MAX__TROPPB'
 'MIN__TROPPB' 'MEAN__TROPPB' 'SUM__TROPPB' 'STD__TROPPB' 'MAX__TROPQ'
 'MIN__TROPQ' 'MEAN__TROPQ' 'SUM__TROPQ' 'STD__TROPQ' 'MAX__TROPT'
 'MIN__TROPT' 'MEAN__TROPT' 'SUM__TROPT' 'STD__TROPT' 'MAX__TS' 'MIN__TS'
 'MEAN__TS' 'SUM__TS' 'STD__TS' 'MAX__TS_MAX' 'MIN__TS_MAX' 'MEAN__TS_MAX'
 'SUM__TS_MAX' 'STD__TS_MAX' 'MAX__TS_MIN' 'MIN__TS_MIN' 'MEAN__TS_MIN'
 'SUM__TS_MIN' 'STD__TS_MIN' 'MAX__TS_RANGE' 'MIN__TS_RANGE'
 'MEAN__TS_RANGE' 'SUM__TS_RANGE' 'STD__TS_RANGE' 'MAX__U10M' 'MIN__U10M'
 'MEAN__U10M' 'SUM__U10M' 'STD__U10M' 'MAX__U2M' 'MIN__U2M' 'MEAN__U2M'
 'SUM__U2M' 'STD__U2M' 'MAX__U50M' 'MIN__U50M' 'MEAN__U50M' 'SUM__U50M'
 'STD__U50M' 'MAX__V10M' 'MIN__V10M' 'MEAN__V10M' 'SUM__V10M' 'STD__V10M'
 'MAX__V2M' 'MIN__V2M' 'MEAN__V2M' 'SUM__V2M' 'STD__V2M' 'MAX__V50M'
 'MIN__V50M' 'MEAN__V50M' 'SUM__V50M' 'STD__V50M' 'MAX__WD10M'
 'MIN__WD10M' 'MEAN__WD10M' 'SUM__WD10M' 'STD__WD10M' 'MAX__WD2M'
 'MIN__WD2M' 'MEAN__WD2M' 'SUM__WD2M' 'STD__WD2M' 'MAX__WD50M'
 'MIN__WD50M' 'MEAN__WD50M' 'SUM__WD50M' 'STD__WD50M' 'MAX__WS10M'
 'MIN__WS10M' 'MEAN__WS10M' 'SUM__WS10M' 'STD__WS10M' 'MAX__WS10M_MAX'
 'MIN__WS10M_MAX' 'MEAN__WS10M_MAX' 'SUM__WS10M_MAX' 'STD__WS10M_MAX'
 'MAX__WS10M_MIN' 'MIN__WS10M_MIN' 'MEAN__WS10M_MIN' 'SUM__WS10M_MIN'
 'STD__WS10M_MIN' 'MAX__WS10M_RANGE' 'MIN__WS10M_RANGE'
 'MEAN__WS10M_RANGE' 'SUM__WS10M_RANGE' 'STD__WS10M_RANGE' 'MAX__WS2M'
 'MIN__WS2M' 'MEAN__WS2M' 'SUM__WS2M' 'STD__WS2M' 'MAX__WS2M_MAX'
 'MIN__WS2M_MAX' 'MEAN__WS2M_MAX' 'SUM__WS2M_MAX' 'STD__WS2M_MAX'
```

```
'MAX__WS2M_MIN' 'MIN__WS2M_MIN' 'MEAN__WS2M_MIN' 'SUM__WS2M_MIN'
'STD__WS2M_MIN' 'MAX__WS2M_RANGE' 'MIN__WS2M_RANGE' 'MEAN__WS2M_RANGE'
'SUM__WS2M_RANGE' 'STD__WS2M_RANGE' 'MAX__WS50M' 'MIN__WS50M'
'MEAN__WS50M' 'SUM__WS50M' 'STD__WS50M' 'MAX__WS50M_MAX' 'MIN__WS50M_MAX'
'MEAN__WS50M_MAX' 'SUM__WS50M_MAX' 'STD__WS50M_MAX' 'MAX__WS50M_MIN'
'MIN__WS50M_MIN' 'MEAN__WS50M_MIN' 'SUM__WS50M_MIN' 'STD__WS50M_MIN'
'MAX__WS50M_RANGE' 'MIN__WS50M_RANGE' 'MEAN__WS50M_RANGE'
'SUM__WS50M_RANGE' 'STD__WS50M_RANGE' 'MAX__Z0M' 'MIN__Z0M' 'MEAN__Z0M'
'SUM__Z0M' 'STD__Z0M']
```

In [484]:
```
print(f"Location: {df_climate['loc'].unique()}")
print(f"Date Range: {str(df_climate['date'].min())} to {str(df_climate['date'].max())}")
```

```
Location: ['BULACAN' 'QUEZON' 'RIZAL']
Date Range: 2016-01-10 to 2021-01-10
```

In [485]:
```
df_climate["date"] = pd.to_datetime(df_climate["date"])
df_climate = df_climate.sort_values("date")
```

In [486]:
```
print(f"Number of Duplicated Rows: {df_climate.duplicated().sum()}")
```

```
Number of Duplicated Rows: 0
```

In [487]:
```
df_climate_bulacan = df_climate[df_climate["loc"] == "BULACAN"]
df_climate_quezon = df_climate[df_climate["loc"] == "QUEZON"]
df_climate_rizal = df_climate[df_climate["loc"] == "RIZAL"]
```

In [488]:
```
df_climate_bulacan = df_climate_bulacan.set_index("date")
df_climate_bulacan = df_climate_bulacan.asfreq("W")
df_climate_bulacan.head()
```

Out[488]:

date	loc	MAX__AIRMASS	MIN__AIRMASS	MEAN__AIRMASS	SUM__AIRMASS	STD__AIRMASS	MAX__ALLSKY_KT	MIN__ALLSKY_KT	MEAN__ALLSKY_KT	SUM__ALLSKY_KT	...	MAX__WS50M_RAN
2016-01-10	BULACAN	-999.0	-999.0	-999.0	-6993.0	0.0	0.71	0.60	0.671429	4.70	...	3
2016-01-17	BULACAN	-999.0	-999.0	-999.0	-6993.0	0.0	0.69	0.61	0.661429	4.63	...	3
2016-01-24	BULACAN	-999.0	-999.0	-999.0	-6993.0	0.0	0.66	0.47	0.612857	4.29	...	5
2016-01-31	BULACAN	-999.0	-999.0	-999.0	-6993.0	0.0	0.63	0.38	0.541429	3.79	...	6
2016-02-07	BULACAN	-999.0	-999.0	-999.0	-6993.0	0.0	0.68	0.20	0.495714	3.47	...	7

5 rows × 501 columns

In [489]:
```
df_climate_quezon = df_climate_quezon.set_index("date")
df_climate_quezon = df_climate_quezon.asfreq("W")
df_climate_quezon.head()
```

Out[489]:

date	loc	MAX__AIRMASS	MIN__AIRMASS	MEAN__AIRMASS	SUM__AIRMASS	STD__AIRMASS	MAX__ALLSKY_KT	MIN__ALLSKY_KT	MEAN__ALLSKY_KT	SUM__ALLSKY_KT	...	MAX__WS50M_RANG
2016-01-10	QUEZON	-999.0	-999.0	-999.0	-6993.0	0.0	0.70	0.48	0.600000	4.20	...	2.
2016-01-17	QUEZON	-999.0	-999.0	-999.0	-6993.0	0.0	0.67	0.40	0.514286	3.60	...	2.
2016-01-24	QUEZON	-999.0	-999.0	-999.0	-6993.0	0.0	0.69	0.25	0.561429	3.93	...	5.
2016-01-31	QUEZON	-999.0	-999.0	-999.0	-6993.0	0.0	0.61	0.38	0.471429	3.30	...	3.
2016-02-07	QUEZON	-999.0	-999.0	-999.0	-6993.0	0.0	0.64	0.23	0.434286	3.04	...	5.

5 rows × 501 columns

In [490]:
```
df_climate_rizal = df_climate_rizal.set_index("date")
df_climate_rizal = df_climate_rizal.asfreq("W")
df_climate_rizal.head()
```

Out[490]:

date	loc	MAX__AIRMASS	MIN__AIRMASS	MEAN__AIRMASS	SUM__AIRMASS	STD__AIRMASS	MAX__ALLSKY_KT	MIN__ALLSKY_KT	MEAN__ALLSKY_KT	SUM__ALLSKY_KT	...	MAX__WS50M_RANGE
2016-01-10	RIZAL	-999.0	-999.0	-999.0	-6993.0	0.0	0.68	0.46	0.562857	3.94	...	2.30
2016-01-17	RIZAL	-999.0	-999.0	-999.0	-6993.0	0.0	0.64	0.47	0.531429	3.72	...	2.65
2016-01-24	RIZAL	-999.0	-999.0	-999.0	-6993.0	0.0	0.66	0.37	0.531429	3.72	...	5.56
2016-01-31	RIZAL	-999.0	-999.0	-999.0	-6993.0	0.0	0.53	0.38	0.448571	3.14	...	4.22
2016-02-07	RIZAL	-999.0	-999.0	-999.0	-6993.0	0.0	0.58	0.20	0.428571	3.00	...	4.10

5 rows × 501 columns

In [491]:
```
display(df_climate_bulacan.describe())
display(df_climate_quezon.describe())
display(df_climate_rizal.describe())
```

	MAX__AIRMASS	MIN__AIRMASS	MEAN__AIRMASS	SUM__AIRMASS	STD__AIRMASS	MAX__ALLSKY_KT	MIN__ALLSKY_KT	MEAN__ALLSKY_KT	SUM__ALLSKY_KT	STD__ALLSKY_KT	...	MAX__WS
count	262.000000	262.000000	262.000000	262.000000	262.000000	262.000000	262.000000	262.000000	262.000000	262.000000	...	
mean	-991.235382	-995.117901	-993.453964	-6954.177748	2.075254	0.638206	0.398931	0.537983	3.765878	0.087024	...	
std	88.699706	62.837266	68.271888	477.903217	33.589798	0.073152	0.160044	0.103445	0.724117	0.047395	...	
min	-999.000000	-999.000000	-999.000000	-6993.000000	0.000000	0.240000	0.050000	0.167143	1.170000	0.008165	...	
25%	-999.000000	-999.000000	-999.000000	-6993.000000	0.000000	0.612500	0.270000	0.478571	3.350000	0.052834	...	
50%	-999.000000	-999.000000	-999.000000	-6993.000000	0.000000	0.660000	0.420000	0.555714	3.890000	0.077766	...	
75%	-999.000000	-999.000000	-999.000000	-6993.000000	0.000000	0.690000	0.520000	0.620000	4.340000	0.113704	...	
max	18.170000	18.110000	18.132857	126.930000	543.698453	0.730000	0.690000	0.702857	4.920000	0.224478	...	

8 rows × 500 columns

	MAX__AIRMASS	MIN__AIRMASS	MEAN__AIRMASS	SUM__AIRMASS	STD__AIRMASS	MAX__ALLSKY_KT	MIN__ALLSKY_KT	MEAN__ALLSKY_KT	SUM__ALLSKY_KT	STD__ALLSKY_KT	...	MAX__WS
count	262.000000	262.000000	262.000000	262.000000	262.000000	262.000000	262.000000	262.000000	262.000000	262.000000	...	
mean	-991.235267	-995.117939	-993.453937	-6954.177557	2.075315	0.622863	0.338511	0.509782	3.568473	0.104508	...	
std	88.701014	62.836648	68.272014	477.904099	33.590458	0.057200	0.153774	0.086393	0.604752	0.050426	...	
min	-999.000000	-999.000000	-999.000000	-6993.000000	0.000000	0.400000	0.050000	0.272857	1.910000	0.009759	...	
25%	-999.000000	-999.000000	-999.000000	-6993.000000	0.000000	0.590000	0.230000	0.446071	3.122500	0.066117	...	
50%	-999.000000	-999.000000	-999.000000	-6993.000000	0.000000	0.630000	0.330000	0.510000	3.570000	0.097784	...	
75%	-999.000000	-999.000000	-999.000000	-6993.000000	0.000000	0.660000	0.467500	0.572500	4.007500	0.141274	...	
max	18.200000	18.100000	18.131429	126.920000	543.709144	0.720000	0.650000	0.677143	4.740000	0.243477	...	

8 rows × 500 columns

	MAX__AIRMASS	MIN__AIRMASS	MEAN__AIRMASS	SUM__AIRMASS	STD__AIRMASS	MAX__ALLSKY_KT	MIN__ALLSKY_KT	MEAN__ALLSKY_KT	SUM__ALLSKY_KT	STD__ALLSKY_KT	...	MAX__WS
count	262.000000	262.000000	262.000000	262.000000	262.000000	262.000000	262.000000	262.000000	262.000000	262.000000	...	
mean	-991.235382	-995.117977	-993.453997	-6954.177977	2.075267	0.589198	0.338664	0.481778	3.372443	0.091041	...	
std	88.699706	62.836030	68.271355	477.899485	33.589908	0.069020	0.131619	0.085031	0.595218	0.040558	...	
min	-999.000000	-999.000000	-999.000000	-6993.000000	0.000000	0.330000	0.040000	0.247143	1.730000	0.011339	...	
25%	-999.000000	-999.000000	-999.000000	-6993.000000	0.000000	0.550000	0.250000	0.415000	2.905000	0.058289	...	
50%	-999.000000	-999.000000	-999.000000	-6993.000000	0.000000	0.600000	0.340000	0.487857	3.415000	0.084219	...	
75%	-999.000000	-999.000000	-999.000000	-6993.000000	0.000000	0.640000	0.440000	0.542143	3.795000	0.117083	...	
max	18.180000	18.090000	18.122857	126.860000	543.700235	0.720000	0.610000	0.652857	4.570000	0.230558	...	

8 rows × 500 columns

In [492]:
```
print(f"Null Values in Bulacan df: \n{df_climate_bulacan[df_climate_bulacan.isnull().any(axis=1)]}\n")
print(f"Null Values in Quezon df: \n{df_climate_quezon[df_climate_quezon.isnull().any(axis=1)]}\n")
print(f"Null Values in Rizal df: \n{df_climate_rizal[df_climate_rizal.isnull().any(axis=1)]}")
```

```
Null Values in Bulacan df:
Empty DataFrame
Columns: [loc, MAX__AIRMASS, MIN__AIRMASS, MEAN__AIRMASS, SUM__AIRMASS, STD__AIRMASS, MAX__ALLSKY_KT, MIN__ALLSKY_KT, MEAN__ALLSKY_KT, SUM__ALLSKY_KT, STD__ALLSKY_KT,
MAX__ALLSKY_SFC_LW_DWN, MIN__ALLSKY_SFC_LW_DWN, MEAN__ALLSKY_SFC_LW_DWN, SUM__ALLSKY_SFC_LW_DWN, STD__ALLSKY_SFC_LW_DWN, MAX__ALLSKY_SFC_LW_UP, MIN__ALLSKY_SFC_LW_UP,
MEAN__ALLSKY_SFC_LW_UP, SUM__ALLSKY_SFC_LW_UP, STD__ALLSKY_SFC_LW_UP, MAX__ALLSKY_SFC_PAR_TOT, MIN__ALLSKY_SFC_PAR_TOT, MEAN__ALLSKY_SFC_PAR_TOT, SUM__ALLSKY_SFC_PAR_
TOT, STD__ALLSKY_SFC_PAR_TOT, MAX__ALLSKY_SFC_SW_DIFF, MIN__ALLSKY_SFC_SW_DIFF, MEAN__ALLSKY_SFC_SW_DIFF, SUM__ALLSKY_SFC_SW_DIFF, STD__ALLSKY_SFC_SW_DIFF, MAX__ALLSK
Y_SFC_SW_DNI, MIN__ALLSKY_SFC_SW_DNI, MEAN__ALLSKY_SFC_SW_DNI, SUM__ALLSKY_SFC_SW_DNI, STD__ALLSKY_SFC_SW_DNI, MAX__ALLSKY_SFC_SW_DWN, MIN__ALLSKY_SFC_SW_DWN, MEAN__A
LLSKY_SFC_SW_DWN, SUM__ALLSKY_SFC_SW_DWN, STD__ALLSKY_SFC_SW_DWN, MAX__ALLSKY_SFC_SW_UP, MIN__ALLSKY_SFC_SW_UP, MEAN__ALLSKY_SFC_SW_UP, SUM__ALLSKY_SFC_SW_UP, STD__AL
LSKY_SFC_SW_UP, MAX__ALLSKY_SFC_UVA, MIN__ALLSKY_SFC_UVA, MEAN__ALLSKY_SFC_UVA, SUM__ALLSKY_SFC_UVA, STD__ALLSKY_SFC_UVA, MAX__ALLSKY_SFC_UVB, MIN__ALLSKY_SFC_UVB, ME
AN__ALLSKY_SFC_UVB, SUM__ALLSKY_SFC_UVB, STD__ALLSKY_SFC_UVB, MAX__ALLSKY_SFC_UV_INDEX, MIN__ALLSKY_SFC_UV_INDEX, MEAN__ALLSKY_SFC_UV_INDEX, SUM__ALLSKY_SFC_UV_INDEX,
STD__ALLSKY_SFC_UV_INDEX, MAX__ALLSKY_SRF_ALB, MIN__ALLSKY_SRF_ALB, MEAN__ALLSKY_SRF_ALB, SUM__ALLSKY_SRF_ALB, STD__ALLSKY_SRF_ALB, MAX__AOD_55, MIN__AOD_55, MEAN__AO
D_55, SUM__AOD_55, STD__AOD_55, MAX__AOD_84, MIN__AOD_84, MEAN__AOD_84, SUM__AOD_84, STD__AOD_84, MAX__CDD0, MIN__CDD0, MEAN__CDD0, SUM__CDD0, STD__CDD0, MAX__CDD10,
MIN__CDD10, MEAN__CDD10, SUM__CDD10, STD__CDD10, MAX__CDD18_3, MIN__CDD18_3, MEAN__CDD18_3, SUM__CDD18_3, STD__CDD18_3, MAX__CLOUD_AMT, MIN__CLOUD_AMT, MEAN__CLOUD_AM
T, SUM__CLOUD_AMT, STD__CLOUD_AMT, MAX__CLOUD_AMT_DAY, MIN__CLOUD_AMT_DAY, MEAN__CLOUD_AMT_DAY, SUM__CLOUD_AMT_DAY, ...]
Index: []

[0 rows x 501 columns]

Null Values in Quezon df:
Empty DataFrame
Columns: [loc, MAX__AIRMASS, MIN__AIRMASS, MEAN__AIRMASS, SUM__AIRMASS, STD__AIRMASS, MAX__ALLSKY_KT, MIN__ALLSKY_KT, MEAN__ALLSKY_KT, SUM__ALLSKY_KT, STD__ALLSKY_KT,
MAX__ALLSKY_SFC_LW_DWN, MIN__ALLSKY_SFC_LW_DWN, MEAN__ALLSKY_SFC_LW_DWN, SUM__ALLSKY_SFC_LW_DWN, STD__ALLSKY_SFC_LW_DWN, MAX__ALLSKY_SFC_LW_UP, MIN__ALLSKY_SFC_LW_UP,
MEAN__ALLSKY_SFC_LW_UP, SUM__ALLSKY_SFC_LW_UP, STD__ALLSKY_SFC_LW_UP, MAX__ALLSKY_SFC_PAR_TOT, MIN__ALLSKY_SFC_PAR_TOT, MEAN__ALLSKY_SFC_PAR_TOT, SUM__ALLSKY_SFC_PAR_
TOT, STD__ALLSKY_SFC_PAR_TOT, MAX__ALLSKY_SFC_SW_DIFF, MIN__ALLSKY_SFC_SW_DIFF, MEAN__ALLSKY_SFC_SW_DIFF, SUM__ALLSKY_SFC_SW_DIFF, STD__ALLSKY_SFC_SW_DIFF, MAX__ALLSK
Y_SFC_SW_DNI, MIN__ALLSKY_SFC_SW_DNI, MEAN__ALLSKY_SFC_SW_DNI, SUM__ALLSKY_SFC_SW_DNI, STD__ALLSKY_SFC_SW_DNI, MAX__ALLSKY_SFC_SW_DWN, MIN__ALLSKY_SFC_SW_DWN, MEAN__A
LLSKY_SFC_SW_DWN, SUM__ALLSKY_SFC_SW_DWN, STD__ALLSKY_SFC_SW_DWN, MAX__ALLSKY_SFC_SW_UP, MIN__ALLSKY_SFC_SW_UP, MEAN__ALLSKY_SFC_SW_UP, SUM__ALLSKY_SFC_SW_UP, STD__AL
LSKY_SFC_SW_UP, MAX__ALLSKY_SFC_UVA, MIN__ALLSKY_SFC_UVA, MEAN__ALLSKY_SFC_UVA, SUM__ALLSKY_SFC_UVA, STD__ALLSKY_SFC_UVA, MAX__ALLSKY_SFC_UVB, MIN__ALLSKY_SFC_UVB, ME
AN__ALLSKY_SFC_UVB, SUM__ALLSKY_SFC_UVB, STD__ALLSKY_SFC_UVB, MAX__ALLSKY_SFC_UV_INDEX, MIN__ALLSKY_SFC_UV_INDEX, MEAN__ALLSKY_SFC_UV_INDEX, SUM__ALLSKY_SFC_UV_INDEX,
STD__ALLSKY_SFC_UV_INDEX, MAX__ALLSKY_SRF_ALB, MIN__ALLSKY_SRF_ALB, MEAN__ALLSKY_SRF_ALB, SUM__ALLSKY_SRF_ALB, STD__ALLSKY_SRF_ALB, MAX__AOD_55, MIN__AOD_55, MEAN__AO
D_55, SUM__AOD_55, STD__AOD_55, MAX__AOD_84, MIN__AOD_84, MEAN__AOD_84, SUM__AOD_84, STD__AOD_84, MAX__CDD0, MIN__CDD0, MEAN__CDD0, SUM__CDD0, STD__CDD0, MAX__CDD10,
MIN__CDD10, MEAN__CDD10, SUM__CDD10, STD__CDD10, MAX__CDD18_3, MIN__CDD18_3, MEAN__CDD18_3, SUM__CDD18_3, STD__CDD18_3, MAX__CLOUD_AMT, MIN__CLOUD_AMT, MEAN__CLOUD_AM
T, SUM__CLOUD_AMT, STD__CLOUD_AMT, MAX__CLOUD_AMT_DAY, MIN__CLOUD_AMT_DAY, MEAN__CLOUD_AMT_DAY, SUM__CLOUD_AMT_DAY, ...]
Index: []

[0 rows x 501 columns]

Null Values in Rizal df:
Empty DataFrame
Columns: [loc, MAX__AIRMASS, MIN__AIRMASS, MEAN__AIRMASS, SUM__AIRMASS, STD__AIRMASS, MAX__ALLSKY_KT, MIN__ALLSKY_KT, MEAN__ALLSKY_KT, SUM__ALLSKY_KT, STD__ALLSKY_KT,
MAX__ALLSKY_SFC_LW_DWN, MIN__ALLSKY_SFC_LW_DWN, MEAN__ALLSKY_SFC_LW_DWN, SUM__ALLSKY_SFC_LW_DWN, STD__ALLSKY_SFC_LW_DWN, MAX__ALLSKY_SFC_LW_UP, MIN__ALLSKY_SFC_LW_UP,
MEAN__ALLSKY_SFC_LW_UP, SUM__ALLSKY_SFC_LW_UP, STD__ALLSKY_SFC_LW_UP, MAX__ALLSKY_SFC_PAR_TOT, MIN__ALLSKY_SFC_PAR_TOT, MEAN__ALLSKY_SFC_PAR_TOT, SUM__ALLSKY_SFC_PAR_
TOT, STD__ALLSKY_SFC_PAR_TOT, MAX__ALLSKY_SFC_SW_DIFF, MIN__ALLSKY_SFC_SW_DIFF, MEAN__ALLSKY_SFC_SW_DIFF, SUM__ALLSKY_SFC_SW_DIFF, STD__ALLSKY_SFC_SW_DIFF, MAX__ALLSK
Y_SFC_SW_DNI, MIN__ALLSKY_SFC_SW_DNI, MEAN__ALLSKY_SFC_SW_DNI, SUM__ALLSKY_SFC_SW_DNI, STD__ALLSKY_SFC_SW_DNI, MAX__ALLSKY_SFC_SW_DWN, MIN__ALLSKY_SFC_SW_DWN, MEAN__A
LLSKY_SFC_SW_DWN, SUM__ALLSKY_SFC_SW_DWN, STD__ALLSKY_SFC_SW_DWN, MAX__ALLSKY_SFC_SW_UP, MIN__ALLSKY_SFC_SW_UP, MEAN__ALLSKY_SFC_SW_UP, SUM__ALLSKY_SFC_SW_UP, STD__AL
LSKY_SFC_SW_UP, MAX__ALLSKY_SFC_UVA, MIN__ALLSKY_SFC_UVA, MEAN__ALLSKY_SFC_UVA, SUM__ALLSKY_SFC_UVA, STD__ALLSKY_SFC_UVA, MAX__ALLSKY_SFC_UVB, MIN__ALLSKY_SFC_UVB, ME
AN__ALLSKY_SFC_UVB, SUM__ALLSKY_SFC_UVB, STD__ALLSKY_SFC_UVB, MAX__ALLSKY_SFC_UV_INDEX, MIN__ALLSKY_SFC_UV_INDEX, MEAN__ALLSKY_SFC_UV_INDEX, SUM__ALLSKY_SFC_UV_INDEX,
STD__ALLSKY_SFC_UV_INDEX, MAX__ALLSKY_SRF_ALB, MIN__ALLSKY_SRF_ALB, MEAN__ALLSKY_SRF_ALB, SUM__ALLSKY_SRF_ALB, STD__ALLSKY_SRF_ALB, MAX__AOD_55, MIN__AOD_55, MEAN__AO
D_55, SUM__AOD_55, STD__AOD_55, MAX__AOD_84, MIN__AOD_84, MEAN__AOD_84, SUM__AOD_84, STD__AOD_84, MAX__CDD0, MIN__CDD0, MEAN__CDD0, SUM__CDD0, STD__CDD0, MAX__CDD10,
MIN__CDD10, MEAN__CDD10, SUM__CDD10, STD__CDD10, MAX__CDD18_3, MIN__CDD18_3, MEAN__CDD18_3, SUM__CDD18_3, STD__CDD18_3, MAX__CLOUD_AMT, MIN__CLOUD_AMT, MEAN__CLOUD_AM
T, SUM__CLOUD_AMT, STD__CLOUD_AMT, MAX__CLOUD_AMT_DAY, MIN__CLOUD_AMT_DAY, MEAN__CLOUD_AMT_DAY, SUM__CLOUD_AMT_DAY, ...]
Index: []

[0 rows x 501 columns]
```

In [493]:
```
df_climate = pd.concat([df_climate_bulacan, df_climate_rizal, df_climate_quezon])
```

In [494]:
```
print(f"Columns: {np.array(df_climate.columns.tolist())}")
```

```
Columns: ['loc' 'MAX__AIRMASS' 'MIN__AIRMASS' 'MEAN__AIRMASS' 'SUM__AIRMASS'
 'STD__AIRMASS' 'MAX__ALLSKY_KT' 'MIN__ALLSKY_KT' 'MEAN__ALLSKY_KT'
 'SUM__ALLSKY_KT' 'STD__ALLSKY_KT' 'MAX__ALLSKY_SFC_LW_DWN'
 'MIN__ALLSKY_SFC_LW_DWN' 'MEAN__ALLSKY_SFC_LW_DWN'
 'SUM__ALLSKY_SFC_LW_DWN' 'STD__ALLSKY_SFC_LW_DWN' 'MAX__ALLSKY_SFC_LW_UP'
 'MIN__ALLSKY_SFC_LW_UP' 'MEAN__ALLSKY_SFC_LW_UP' 'SUM__ALLSKY_SFC_LW_UP'
 'STD__ALLSKY_SFC_LW_UP' 'MAX__ALLSKY_SFC_PAR_TOT'
 'MIN__ALLSKY_SFC_PAR_TOT' 'MEAN__ALLSKY_SFC_PAR_TOT'
 'SUM__ALLSKY_SFC_PAR_TOT' 'STD__ALLSKY_SFC_PAR_TOT'
 'MAX__ALLSKY_SFC_SW_DIFF' 'MIN__ALLSKY_SFC_SW_DIFF'
 'MEAN__ALLSKY_SFC_SW_DIFF' 'SUM__ALLSKY_SFC_SW_DIFF'
 'STD__ALLSKY_SFC_SW_DIFF' 'MAX__ALLSKY_SFC_SW_DNI'
 'MIN__ALLSKY_SFC_SW_DNI' 'MEAN__ALLSKY_SFC_SW_DNI'
 'SUM__ALLSKY_SFC_SW_DNI' 'STD__ALLSKY_SFC_SW_DNI'
 'MAX__ALLSKY_SFC_SW_DWN' 'MIN__ALLSKY_SFC_SW_DWN'
 'MEAN__ALLSKY_SFC_SW_DWN' 'SUM__ALLSKY_SFC_SW_DWN'
 'STD__ALLSKY_SFC_SW_DWN' 'MAX__ALLSKY_SFC_SW_UP' 'MIN__ALLSKY_SFC_SW_UP'
 'MEAN__ALLSKY_SFC_SW_UP' 'SUM__ALLSKY_SFC_SW_UP' 'STD__ALLSKY_SFC_SW_UP'
 'MAX__ALLSKY_SFC_UVA' 'MIN__ALLSKY_SFC_UVA' 'MEAN__ALLSKY_SFC_UVA'
 'SUM__ALLSKY_SFC_UVA' 'STD__ALLSKY_SFC_UVA' 'MAX__ALLSKY_SFC_UVB'
 'MIN__ALLSKY_SFC_UVB' 'MEAN__ALLSKY_SFC_UVB' 'SUM__ALLSKY_SFC_UVB'
 'STD__ALLSKY_SFC_UVB' 'MAX__ALLSKY_SFC_UV_INDEX'
 'MIN__ALLSKY_SFC_UV_INDEX' 'MEAN__ALLSKY_SFC_UV_INDEX'
 'SUM__ALLSKY_SFC_UV_INDEX' 'STD__ALLSKY_SFC_UV_INDEX'
 'MAX__ALLSKY_SRF_ALB' 'MIN__ALLSKY_SRF_ALB' 'MEAN__ALLSKY_SRF_ALB'
 'SUM__ALLSKY_SRF_ALB' 'STD__ALLSKY_SRF_ALB' 'MAX__AOD_55' 'MIN__AOD_55'
 'MEAN__AOD_55' 'SUM__AOD_55' 'STD__AOD_55' 'MAX__AOD_84' 'MIN__AOD_84'
 'MEAN__AOD_84' 'SUM__AOD_84' 'STD__AOD_84' 'MAX__CDD0' 'MIN__CDD0'
 'MEAN__CDD0' 'SUM__CDD0' 'STD__CDD0' 'MAX__CDD10' 'MIN__CDD10'
 'MEAN__CDD10' 'SUM__CDD10' 'STD__CDD10' 'MAX__CDD18_3' 'MIN__CDD18_3'
 'MEAN__CDD18_3' 'SUM__CDD18_3' 'STD__CDD18_3' 'MAX__CLOUD_AMT'
 'MIN__CLOUD_AMT' 'MEAN__CLOUD_AMT' 'SUM__CLOUD_AMT' 'STD__CLOUD_AMT'
 'MAX__CLOUD_AMT_DAY' 'MIN__CLOUD_AMT_DAY' 'MEAN__CLOUD_AMT_DAY'
 'SUM__CLOUD_AMT_DAY' 'STD__CLOUD_AMT_DAY' 'MAX__CLOUD_AMT_NIGHT'
 'MIN__CLOUD_AMT_NIGHT' 'MEAN__CLOUD_AMT_NIGHT' 'SUM__CLOUD_AMT_NIGHT'
 'STD__CLOUD_AMT_NIGHT' 'MAX__CLOUD_OD' 'MIN__CLOUD_OD' 'MEAN__CLOUD_OD'
 'SUM__CLOUD_OD' 'STD__CLOUD_OD' 'MAX__CLRSKY_DAYS' 'MIN__CLRSKY_DAYS'
 'MEAN__CLRSKY_DAYS' 'SUM__CLRSKY_DAYS' 'STD__CLRSKY_DAYS'
 'MAX__CLRSKY_KT' 'MIN__CLRSKY_KT' 'MEAN__CLRSKY_KT' 'SUM__CLRSKY_KT'
 'STD__CLRSKY_KT' 'MAX__CLRSKY_SFC_LW_DWN' 'MIN__CLRSKY_SFC_LW_DWN'
 'MEAN__CLRSKY_SFC_LW_DWN' 'SUM__CLRSKY_SFC_LW_DWN'
 'STD__CLRSKY_SFC_LW_DWN' 'MAX__CLRSKY_SFC_LW_UP' 'MIN__CLRSKY_SFC_LW_UP'
 'MEAN__CLRSKY_SFC_LW_UP' 'SUM__CLRSKY_SFC_LW_UP' 'STD__CLRSKY_SFC_LW_UP'
 'MAX__CLRSKY_SFC_PAR_TOT' 'MIN__CLRSKY_SFC_PAR_TOT'
 'MEAN__CLRSKY_SFC_PAR_TOT' 'SUM__CLRSKY_SFC_PAR_TOT'
 'STD__CLRSKY_SFC_PAR_TOT' 'MAX__CLRSKY_SFC_SW_DIFF'
 'MIN__CLRSKY_SFC_SW_DIFF' 'MEAN__CLRSKY_SFC_SW_DIFF'
 'SUM__CLRSKY_SFC_SW_DIFF' 'STD__CLRSKY_SFC_SW_DIFF'
 'MAX__CLRSKY_SFC_SW_DNI' 'MIN__CLRSKY_SFC_SW_DNI'
 'MEAN__CLRSKY_SFC_SW_DNI' 'SUM__CLRSKY_SFC_SW_DNI'
 'STD__CLRSKY_SFC_SW_DNI' 'MAX__CLRSKY_SFC_SW_DWN'
 'MIN__CLRSKY_SFC_SW_DWN' 'MEAN__CLRSKY_SFC_SW_DWN'
 'SUM__CLRSKY_SFC_SW_DWN' 'STD__CLRSKY_SFC_SW_DWN' 'MAX__CLRSKY_SFC_SW_UP'
 'MIN__CLRSKY_SFC_SW_UP' 'MEAN__CLRSKY_SFC_SW_UP' 'SUM__CLRSKY_SFC_SW_UP'
 'STD__CLRSKY_SFC_SW_UP' 'MAX__CLRSKY_SRF_ALB' 'MIN__CLRSKY_SRF_ALB'
 'MEAN__CLRSKY_SRF_ALB' 'SUM__CLRSKY_SRF_ALB' 'STD__CLRSKY_SRF_ALB'
 'MAX__DISPH' 'MIN__DISPH' 'MEAN__DISPH' 'SUM__DISPH' 'STD__DISPH'
 'MAX__EVLAND' 'MIN__EVLAND' 'MEAN__EVLAND' 'SUM__EVLAND' 'STD__EVLAND'
 'MAX__EVPTRNS' 'MIN__EVPTRNS' 'MEAN__EVPTRNS' 'SUM__EVPTRNS'
 'STD__EVPTRNS' 'MAX__FROST_DAYS' 'MIN__FROST_DAYS' 'MEAN__FROST_DAYS'
 'SUM__FROST_DAYS' 'STD__FROST_DAYS' 'MAX__FRSEAICE' 'MIN__FRSEAICE'
 'MEAN__FRSEAICE' 'SUM__FRSEAICE' 'STD__FRSEAICE' 'MAX__FRSNO'
 'MIN__FRSNO' 'MEAN__FRSNO' 'SUM__FRSNO' 'STD__FRSNO' 'MAX__GWETPROF'
 'MIN__GWETPROF' 'MEAN__GWETPROF' 'SUM__GWETPROF' 'STD__GWETPROF'
 'MAX__GWETROOT' 'MIN__GWETROOT' 'MEAN__GWETROOT' 'SUM__GWETROOT'
 'STD__GWETROOT' 'MAX__GWETTOP' 'MIN__GWETTOP' 'MEAN__GWETTOP'
 'SUM__GWETTOP' 'STD__GWETTOP' 'MAX__HDD0' 'MIN__HDD0' 'MEAN__HDD0'
 'SUM__HDD0' 'STD__HDD0' 'MAX__HDD10' 'MIN__HDD10' 'MEAN__HDD10'
 'SUM__HDD10' 'STD__HDD10' 'MAX__HDD18_3' 'MIN__HDD18_3' 'MEAN__HDD18_3'
 'SUM__HDD18_3' 'STD__HDD18_3' 'MAX__MIDDAY_INSOL' 'MIN__MIDDAY_INSOL'
 'MEAN__MIDDAY_INSOL' 'SUM__MIDDAY_INSOL' 'STD__MIDDAY_INSOL'
 'MAX__PBLTOP' 'MIN__PBLTOP' 'MEAN__PBLTOP' 'SUM__PBLTOP' 'STD__PBLTOP'
 'MAX__PRECSNOLAND' 'MIN__PRECSNOLAND' 'MEAN__PRECSNOLAND'
 'SUM__PRECSNOLAND' 'STD__PRECSNOLAND' 'MAX__PRECTOTCORR'
 'MIN__PRECTOTCORR' 'MEAN__PRECTOTCORR' 'SUM__PRECTOTCORR'
 'STD__PRECTOTCORR' 'MAX__PS' 'MIN__PS' 'MEAN__PS' 'SUM__PS' 'STD__PS'
 'MAX__PW' 'MIN__PW' 'MEAN__PW' 'SUM__PW' 'STD__PW' 'MAX__QV10M'
 'MIN__QV10M' 'MEAN__QV10M' 'SUM__QV10M' 'STD__QV10M' 'MAX__QV2M'
 'MIN__QV2M' 'MEAN__QV2M' 'SUM__QV2M' 'STD__QV2M' 'MAX__RH2M' 'MIN__RH2M'
 'MEAN__RH2M' 'SUM__RH2M' 'STD__RH2M' 'MAX__RHOA' 'MIN__RHOA' 'MEAN__RHOA'
 'SUM__RHOA' 'STD__RHOA' 'MAX__SG_NOON' 'MIN__SG_NOON' 'MEAN__SG_NOON'
 'SUM__SG_NOON' 'STD__SG_NOON' 'MAX__SLP' 'MIN__SLP' 'MEAN__SLP'
 'SUM__SLP' 'STD__SLP' 'MAX__SNODP' 'MIN__SNODP' 'MEAN__SNODP'
 'SUM__SNODP' 'STD__SNODP' 'MAX__T10M' 'MIN__T10M' 'MEAN__T10M'
 'SUM__T10M' 'STD__T10M' 'MAX__T10M_MAX' 'MIN__T10M_MAX' 'MEAN__T10M_MAX'
 'SUM__T10M_MAX' 'STD__T10M_MAX' 'MAX__T10M_MIN' 'MIN__T10M_MIN'
 'MEAN__T10M_MIN' 'SUM__T10M_MIN' 'STD__T10M_MIN' 'MAX__T10M_RANGE'
 'MIN__T10M_RANGE' 'MEAN__T10M_RANGE' 'SUM__T10M_RANGE' 'STD__T10M_RANGE'
 'MAX__T2M' 'MIN__T2M' 'MEAN__T2M' 'SUM__T2M' 'STD__T2M' 'MAX__T2MDEW'
 'MIN__T2MDEW' 'MEAN__T2MDEW' 'SUM__T2MDEW' 'STD__T2MDEW' 'MAX__T2MWET'
 'MIN__T2MWET' 'MEAN__T2MWET' 'SUM__T2MWET' 'STD__T2MWET' 'MAX__T2M_MAX'
 'MIN__T2M_MAX' 'MEAN__T2M_MAX' 'SUM__T2M_MAX' 'STD__T2M_MAX'
 'MAX__T2M_MIN' 'MIN__T2M_MIN' 'MEAN__T2M_MIN' 'SUM__T2M_MIN'
 'STD__T2M_MIN' 'MAX__T2M_RANGE' 'MIN__T2M_RANGE' 'MEAN__T2M_RANGE'
 'SUM__T2M_RANGE' 'STD__T2M_RANGE' 'MAX__TO3' 'MIN__TO3' 'MEAN__TO3'
 'SUM__TO3' 'STD__TO3' 'MAX__TOA_SW_DNI' 'MIN__TOA_SW_DNI'
 'MEAN__TOA_SW_DNI' 'SUM__TOA_SW_DNI' 'STD__TOA_SW_DNI' 'MAX__TOA_SW_DWN'
 'MIN__TOA_SW_DWN' 'MEAN__TOA_SW_DWN' 'SUM__TOA_SW_DWN' 'STD__TOA_SW_DWN'
 'MAX__TQV' 'MIN__TQV' 'MEAN__TQV' 'SUM__TQV' 'STD__TQV' 'MAX__TROPPB'
 'MIN__TROPPB' 'MEAN__TROPPB' 'SUM__TROPPB' 'STD__TROPPB' 'MAX__TROPQ'
 'MIN__TROPQ' 'MEAN__TROPQ' 'SUM__TROPQ' 'STD__TROPQ' 'MAX__TROPT'
 'MIN__TROPT' 'MEAN__TROPT' 'SUM__TROPT' 'STD__TROPT' 'MAX__TS' 'MIN__TS'
 'MEAN__TS' 'SUM__TS' 'STD__TS' 'MAX__TS_MAX' 'MIN__TS_MAX' 'MEAN__TS_MAX'
 'SUM__TS_MAX' 'STD__TS_MAX' 'MAX__TS_MIN' 'MIN__TS_MIN' 'MEAN__TS_MIN'
 'SUM__TS_MIN' 'STD__TS_MIN' 'MAX__TS_RANGE' 'MIN__TS_RANGE'
 'MEAN__TS_RANGE' 'SUM__TS_RANGE' 'STD__TS_RANGE' 'MAX__U10M' 'MIN__U10M'
 'MEAN__U10M' 'SUM__U10M' 'STD__U10M' 'MAX__U2M' 'MIN__U2M' 'MEAN__U2M'
 'SUM__U2M' 'STD__U2M' 'MAX__U50M' 'MIN__U50M' 'MEAN__U50M' 'SUM__U50M'
 'STD__U50M' 'MAX__V10M' 'MIN__V10M' 'MEAN__V10M' 'SUM__V10M' 'STD__V10M'
 'MAX__V2M' 'MIN__V2M' 'MEAN__V2M' 'SUM__V2M' 'STD__V2M' 'MAX__V50M'
 'MIN__V50M' 'MEAN__V50M' 'SUM__V50M' 'STD__V50M' 'MAX__WD10M'
 'MIN__WD10M' 'MEAN__WD10M' 'SUM__WD10M' 'STD__WD10M' 'MAX__WD2M'
 'MIN__WD2M' 'MEAN__WD2M' 'SUM__WD2M' 'STD__WD2M' 'MAX__WD50M'
 'MIN__WD50M' 'MEAN__WD50M' 'SUM__WD50M' 'STD__WD50M' 'MAX__WS10M'
 'MIN__WS10M' 'MEAN__WS10M' 'SUM__WS10M' 'STD__WS10M' 'MAX__WS10M_MAX'
 'MIN__WS10M_MAX' 'MEAN__WS10M_MAX' 'SUM__WS10M_MAX' 'STD__WS10M_MAX'
 'MAX__WS10M_MIN' 'MIN__WS10M_MIN' 'MEAN__WS10M_MIN' 'SUM__WS10M_MIN'
 'STD__WS10M_MIN' 'MAX__WS10M_RANGE' 'MIN__WS10M_RANGE'
 'MEAN__WS10M_RANGE' 'SUM__WS10M_RANGE' 'STD__WS10M_RANGE' 'MAX__WS2M'
 'MIN__WS2M' 'MEAN__WS2M' 'SUM__WS2M' 'STD__WS2M' 'MAX__WS2M_MAX'
 'MIN__WS2M_MAX' 'MEAN__WS2M_MAX' 'SUM__WS2M_MAX' 'STD__WS2M_MAX'
 'MAX__WS2M_MIN' 'MIN__WS2M_MIN' 'MEAN__WS2M_MIN' 'SUM__WS2M_MIN'
```

```
'STD__WS2M_MIN' 'MAX__WS2M_RANGE' 'MIN__WS2M_RANGE' 'MEAN__WS2M_RANGE'
'SUM__WS2M_RANGE' 'STD__WS2M_RANGE' 'MAX__WS50M' 'MIN__WS50M'
'MEAN__WS50M' 'SUM__WS50M' 'STD__WS50M' 'MAX__WS50M_MAX' 'MIN__WS50M_MAX'
'MEAN__WS50M_MAX' 'SUM__WS50M_MAX' 'STD__WS50M_MAX' 'MAX__WS50M_MIN'
'MIN__WS50M_MIN' 'MEAN__WS50M_MIN' 'SUM__WS50M_MIN' 'STD__WS50M_MIN'
'MAX__WS50M_RANGE' 'MIN__WS50M_RANGE' 'MEAN__WS50M_RANGE'
'SUM__WS50M_RANGE' 'STD__WS50M_RANGE' 'MAX__Z0M' 'MIN__Z0M' 'MEAN__Z0M'
'SUM__Z0M' 'STD__Z0M']
```

clou_amt

1. **CLOUD_AMT(%, recom, mean)**: The average percent of cloud amount during the temporal period.
2. **CLOUD_AMT_DAY(%, neut, mean)**: The average percent of cloud amount during daylight.
3. **CLOUD_AMT_NIGHT(%, neut, mean)**: The average percent of cloud amount during nighttime.

precip measure of the water that has already condensed and fallen to the surface)

1. **PRECTOTCORR(mm/day, recom, mean)**:

- The bias corrected average of total precipitation at the surface of the earth in water mass (includes water content in snow).
- This measures the average total precipitation, including both rain and the water content in snow, that falls to the Earth's surface. The 'bias corrected' part indicates that adjustments have been made to the data to account for any systematic errors in the measurement process. This ensures that the data accurately reflects the actual amount of precipitation.

water_vapor (gaseous phase in the atmosphere)

1. **PW(cm, neut, mean)**:

- The total atmospheric water vapor contained in a vertical column of the atmosphere.
- This is the total amount of water vapor present in the entire vertical column of the atmosphere. It's a measure of the total water content in the air above a specific location, from the ground up to the top of the atmosphere.

2. **TQV(kg m-2, recom, mean)**:

- The total atmospheric water vapor contained in a vertical column of unit cross-sectional area extending from the surface to the top of the atmosphere.
- This is specifically measured in a vertical column of unit cross-sectional area. It's often expressed in terms of the height to which that water substance would stand if completely condensed and collected in a vessel of the same unit cross section. This is also known as precipitable water vapor and is an important parameter for characterizing the water vapor content in the atmosphere.

specific_humidity

1. **QV10M(g water/kg air, recom, mean)**:

- The ratio of the mass of water vapor to the total mass of air at 10 meters (g water/kg total air)

2. **QV2M(g water/kg air, recom, mean)**:

- The ratio of the mass of water vapor to the total mass of air at 2 meters (g water/kg total air).

3. **TROPQ(g water/kg air, neut, mean)**:

- The ratio of the mass of water vapor to the total mass of air at the tropopause pressure given by TROPPB (g water/kg total air).

relative_humidity

1. **RH2M(%, recom, mean)**:

- The ratio of actual partial pressure of water vapor to the partial pressure at saturation, expressed in percent.

temperature

1. **T2M: Average Air (Dry Bulb) Temperature at 2 Meters**:

- This is the standard height for measuring near-surface air temperature, which represents the average temperature we experience in daily life. It's measured using a thermometer placed in a shaded area, typically in a Stevenson screen, to avoid direct sunlight affecting the reading[1].

2. **T10M: Air (Dry Bulb) Temperature at 10 Meters**:

- This measurement is taken at a higher altitude and can be different from the 2-meter temperature due to factors like wind speed, surface heating, and atmospheric conditions. It's less influenced by the immediate ground surface and more representative of the air mass characteristics[1].

3. **TS: Average Temperature at the Earth's Surface**: This is a measure of the actual temperature of the ground or surface materials like soil, vegetation, or pavement. It's influenced by solar radiation, thermal properties of the surface, and ambient air temperatures[1].
4. **TROPT: Air (Dry Bulb) Temperature at the Tropopause**: The tropopause is the boundary layer between the troposphere and the stratosphere. Temperature measurements here are significant for understanding weather patterns and the behavior of the atmosphere at higher altitudes. The temperature at the tropopause is generally much lower than near the surface[1].
5. **T2MDEW: Dew/Frost Point Temperature at 2 Meters**: The dew point is the temperature at which air becomes saturated with moisture and dew forms. The frost point is similar but for the formation of frost. This measurement is crucial for understanding humidity and the potential for dew or frost formation near the ground[1].
6. **T2MWET: Adiabatic Saturation Temperature**: This is measured by a wet-bulb thermometer, which has a water-soaked cloth wrapped around the bulb. Air passing over the wet bulb causes evaporation, cooling the thermometer. This temperature is important for determining the thermodynamic properties of moist air and is used in calculations like heat index and humidex[1].

THERMAL IRRADIANCE

Note: We exclude clear sky and use all sky

1. **ALLSKY_SFC_LW_DWN: Downward Thermal Infrared Irradiance**

- The downward thermal infrared irradiance under all sky conditions reaching a horizontal plane the surface of the earth. Also known as Horizontal Infrared Radiation Intensity from Sky.
- This measures the infrared radiation coming from the sky to the Earth's surface. It's significant for understanding the Earth's heat balance and is also known as Horizontal Infrared Radiation Intensity from Sky.

2. **ALLSKY_SFC_LW_UP: Upward Thermal Infrared Irradiance**:

- The upward thermal infrared irradiance under all sky conditions.
- This is the infrared radiation that is emitted upwards from the Earth's surface into the atmosphere. It's important for calculating the Earth's radiation budget.

SOLAR IRRADIANCE

1. **ALLSKY_SFC_SW_DIFF: Diffuse Solar Irradiance**:

- The diffuse (light energy scattered out of the direction of the sun) solar irradiance incident on a horizontal plane at the surface of the earth under all sky conditions.
- This is sunlight that has been scattered by molecules and particles in the atmosphere, spreading out of the direct path from the sun and reaching the Earth's surface.

2. **ALLSKY_SFC_SW_DNI: Direct Solar Irradiance**:

- The direct solar irradiance incident to a horizontal plane normal (perpendicular) to the direction of the sun's position under all sky conditions.
- This measures the sunlight that travels in a straight line from the sun to the Earth's surface without being scattered by the atmosphere. It's measured on a surface that is perpendicular to the direction of the sun.

3. **ALLSKY_SFC_SW_DWN: Total Solar Irradiance (Global Horizontal Irradiance, GHI)**:

- The total solar irradiance incident (direct plus diffuse) on a horizontal plane at the surface of the earth under all sky conditions. An alternative term for the total solar irradiance is the "Global Horizontal Irradiance" or GHI.
- GHI is the sum of direct and diffuse solar irradiance on a horizontal surface. It's a key parameter for the design and evaluation of solar energy systems.

4. **ALLSKY_SFC_SW_UP: Upward Shortwave Irradiance**:

- The upward shortwave irradiance under all sky conditions.
- This parameter measures the shortwave radiation (part of the solar spectrum) that is reflected or scattered back into the atmosphere from the Earth's surface.

5. **MIDDAY_INSOL: Solar Noon Hour Midday Solar Irradiance**: This measures the total solar irradiance (direct plus diffuse) reaching a horizontal plane at the Earth's surface during the midday period around solar noon. It's a peak period for solar energy availability.

ACTIVE RADIATION

1. **Total Photosynthetically Active Radiation (PAR)**:

- The total Photosynthetically Active Radiation (PAR) incident on a horizontal plane at the surface of the earth under all sky conditions.
- PAR represents the range of light wavelengths that plants use for photosynthesis. This parameter measures the total amount of this light that reaches the Earth's surface.

UV IRRADIANCE

1. **ALLSKY_SFC_UVA: Ultraviolet A (UVA) Irradiance**: UVA irradiance measures the intensity of UVA rays, which are the longer wavelengths of ultraviolet light and can penetrate deeper into the skin.
2. **ALLSKY_SFC_UVB: Ultraviolet B (UVB) Irradiance**: UVB irradiance measures the intensity of UVB rays, which have shorter wavelengths and are primarily responsible for causing sunburn.

EXTRATERRESTRIAL RADIATION

1. **TOA_SW_DNI: Extraterrestrial Radiation on a Plane Oriented to the Sun**: This is the total solar irradiance (direct plus diffuse) incident on a plane that is oriented perpendicular to the direction of the sun's rays at the top of the atmosphere. It represents the maximum potential solar radiation a location can receive without atmospheric interference.
2. **TOA_SW_DWN: Extraterrestrial Radiation on a Horizontal Plane**: Similar to the previous parameter, this measures the total solar irradiance at the top of the atmosphere, but on a horizontal plane. It's used as a reference for calculating the Earth's albedo and for understanding the solar energy input into the Earth's climate system².

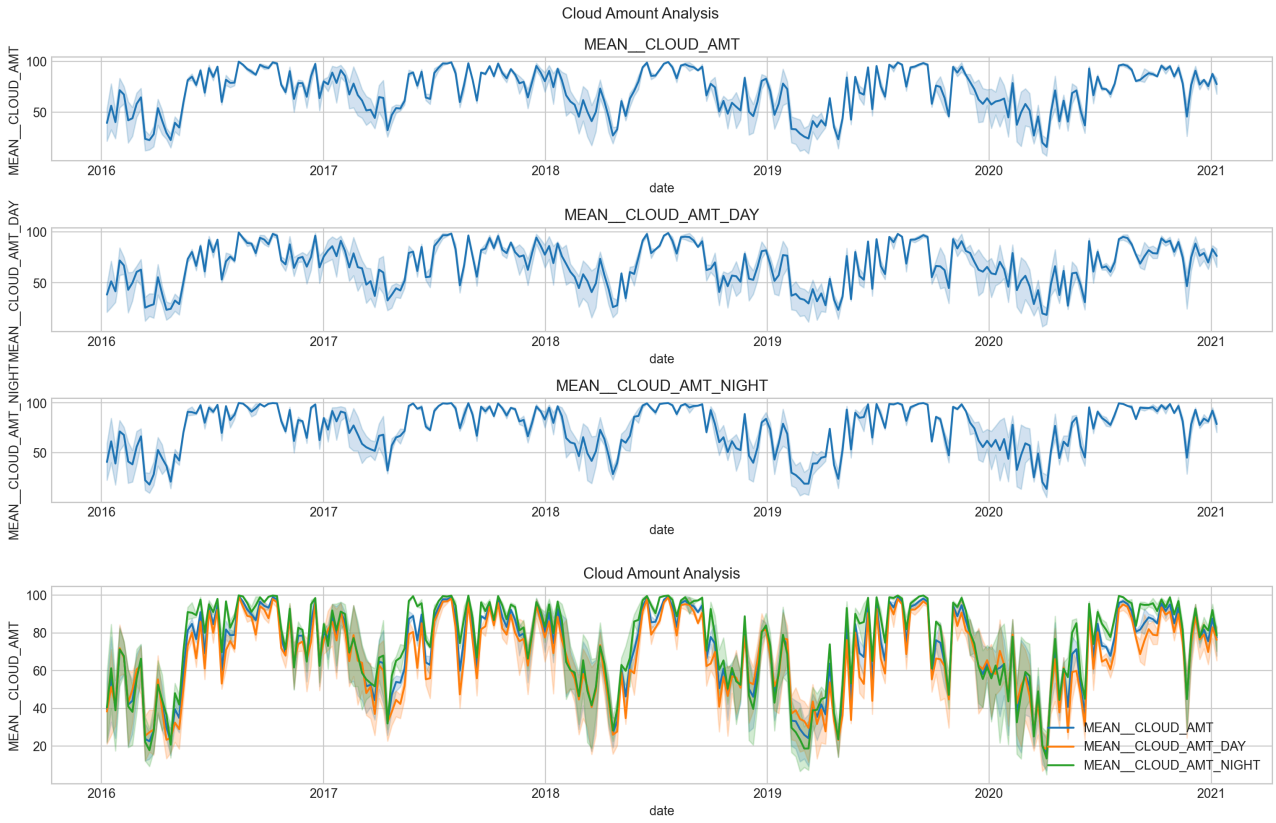
EVAPOTRANSPIRATION

1. **EVPTRNS: Evapotranspiration Energy Flux**: This refers to the combined process of water evaporation from the Earth's surface and transpiration from plants. It's a critical component in the hydrological cycle and is influenced by factors like solar radiation, temperature, humidity, and wind¹.

In [495]:
```
cloud_amt_ = ['MEAN__CLOUD_AMT', 'MEAN__CLOUD_AMT_DAY', 'MEAN__CLOUD_AMT_NIGHT'] # in %
precip_ = ['SUM__PRECTOTCORR', 'MEAN__PRECTOTCORR'] # in mm/day
water_vapor_ = ['SUM__PW', 'MEAN__TQV']
specific_humidity_ = ['MEAN__QV10M', 'MEAN__QV2M']
relative_humidity_ = ['MEAN__RH2M'] # in %
soil_moisture_ = ['MEAN__GWETPROF', 'MEAN__GWETROOT', 'MEAN__GWETTOP'] # in 1

temperature_mean_ = ['MEAN__T2M', 'MEAN__T10M', 'MEAN__TS', # for ave daily
                     'MEAN__T2MDEW', 'MEAN__T2MWET']
temperature_max_ = ['MEAN__T2M_MAX', 'MEAN__T10M_MAX', 'MEAN__TS_MAX', # for max daily
                    'MAX__T2M_MAX',   'MAX__T10M_MAX',  'MAX__TS_MAX']
temperature_min_ = ['MEAN__T2M_MIN', 'MEAN__T10M_MIN', 'MEAN__TS_MIN', # for min daily
                'MIN__T2M_MIN',  'MIN__T10M_MIN',  'MIN__TS_MIN']
temperature_range_ = ['MEAN__T2M_RANGE', 'MEAN__T10M_RANGE', 'MEAN__TS_RANGE'] # for range

others = ['MEAN__ALLSKY_SFC_PAR_TOT','MEAN__EVPTRNS',]
irradiance_lw = ['MEAN__ALLSKY_SFC_LW_UP','MEAN__ALLSKY_SFC_LW_DWN']
irradiance_sw = ['MEAN__MIDDAY_INSOL', 'MEAN__ALLSKY_SFC_SW_DIFF', 'MEAN__ALLSKY_SFC_SW_DNI', 'MEAN__ALLSKY_SFC_SW_DWN', 'MEAN__ALLSKY_SFC_SW_UP']
irradiance_uv = ['MEAN__ALLSKY_SFC_UVA', 'MEAN__ALLSKY_SFC_UVB']
irradiance_toa = ['MEAN__TOA_SW_DNI', 'MEAN__TOA_SW_DWN']
```

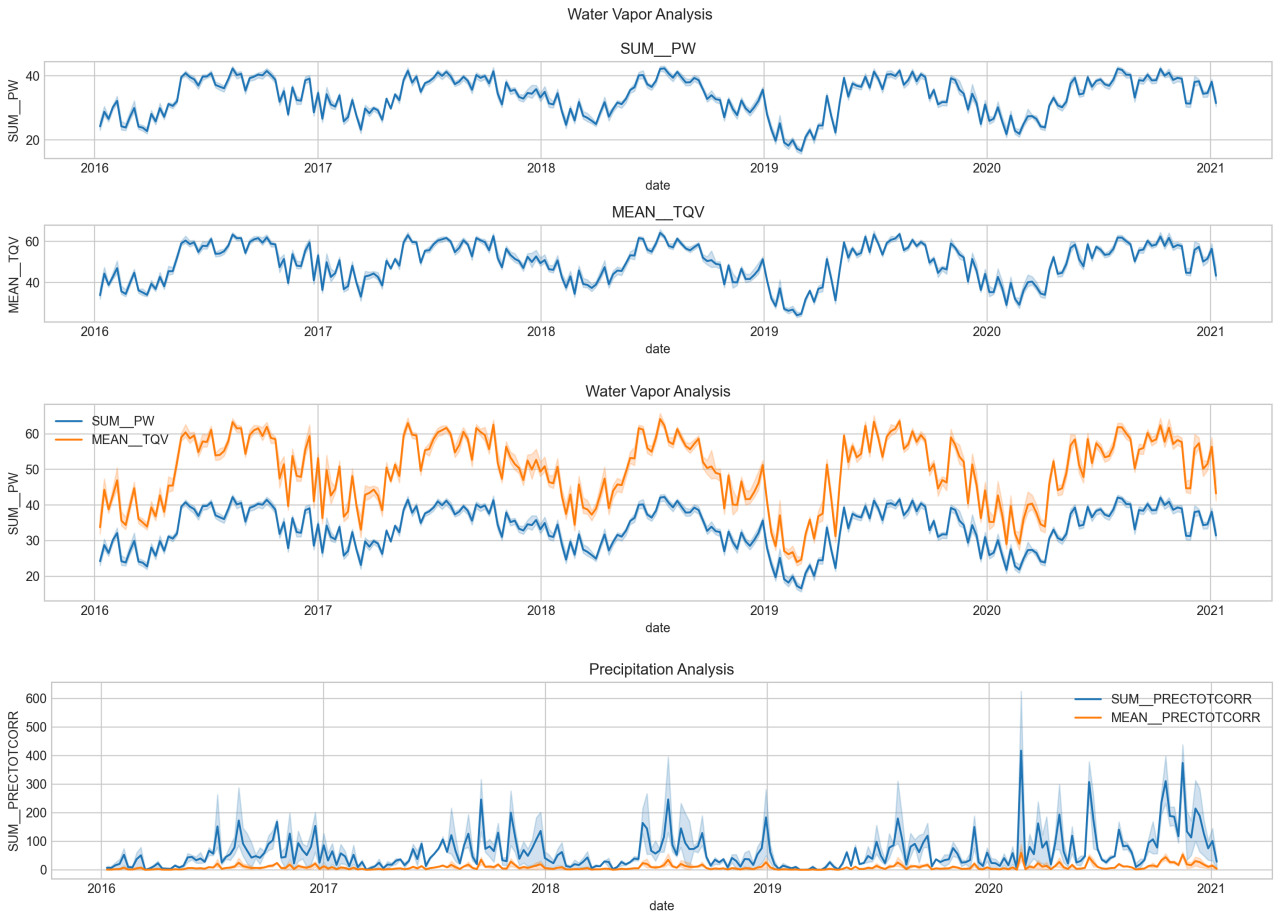
In [496]:
```
def climate_plot(column_names, title):
    num_plots = len(column_names)
    fig, axs = plt.subplots(num_plots, figsize=(14, num_plots*2))

    if num_plots == 1:
        axs = [axs]

    if num_plots == 2:
        axs = axs.flatten()

    for i, column in enumerate(column_names):
        sns.lineplot(x=df_climate.index, y=df_climate[column], ax=axs[i])
        axs[i].set_title(column)

    plt.suptitle(title)
    plt.tight_layout()
    plt.show()

def climate_plot_combine(column_names, title):
    fig, ax = plt.subplots(figsize=(14, 3))

    for i, column in enumerate(column_names):
        sns.lineplot(x=df_climate.index, y=df_climate[column], ax=ax, label=column)

    plt.title(title)
    plt.tight_layout()
    plt.show()
```

In [497]:
```
climate_plot(cloud_amt_, 'Cloud Amount Analysis')
climate_plot_combine(cloud_amt_, 'Cloud Amount Analysis')
```

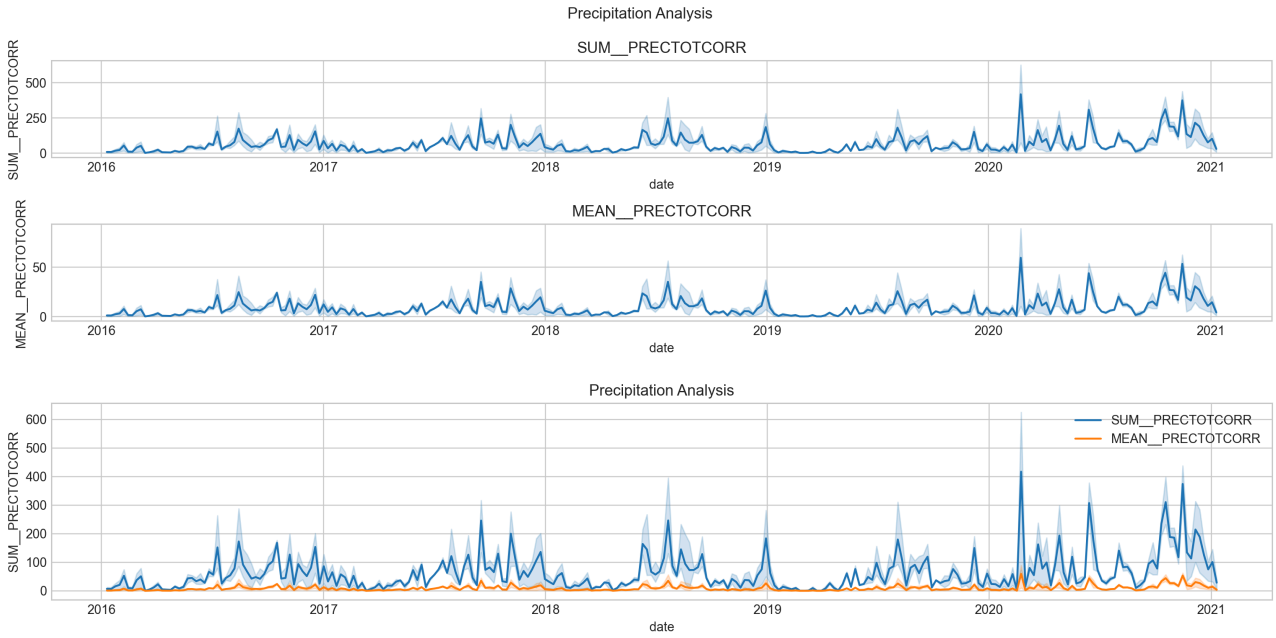


In [498]:
```
cloud_amt = ['MEAN__CLOUD_AMT'] # Removed MEAN_CLOUD_AMT_DAY and MEAN_CLOUD_AMT_NIGHT
```

In [499]:
```
climate_plot(water_vapor_, 'Water Vapor Analysis')
climate_plot_combine(water_vapor_, 'Water Vapor Analysis')
```

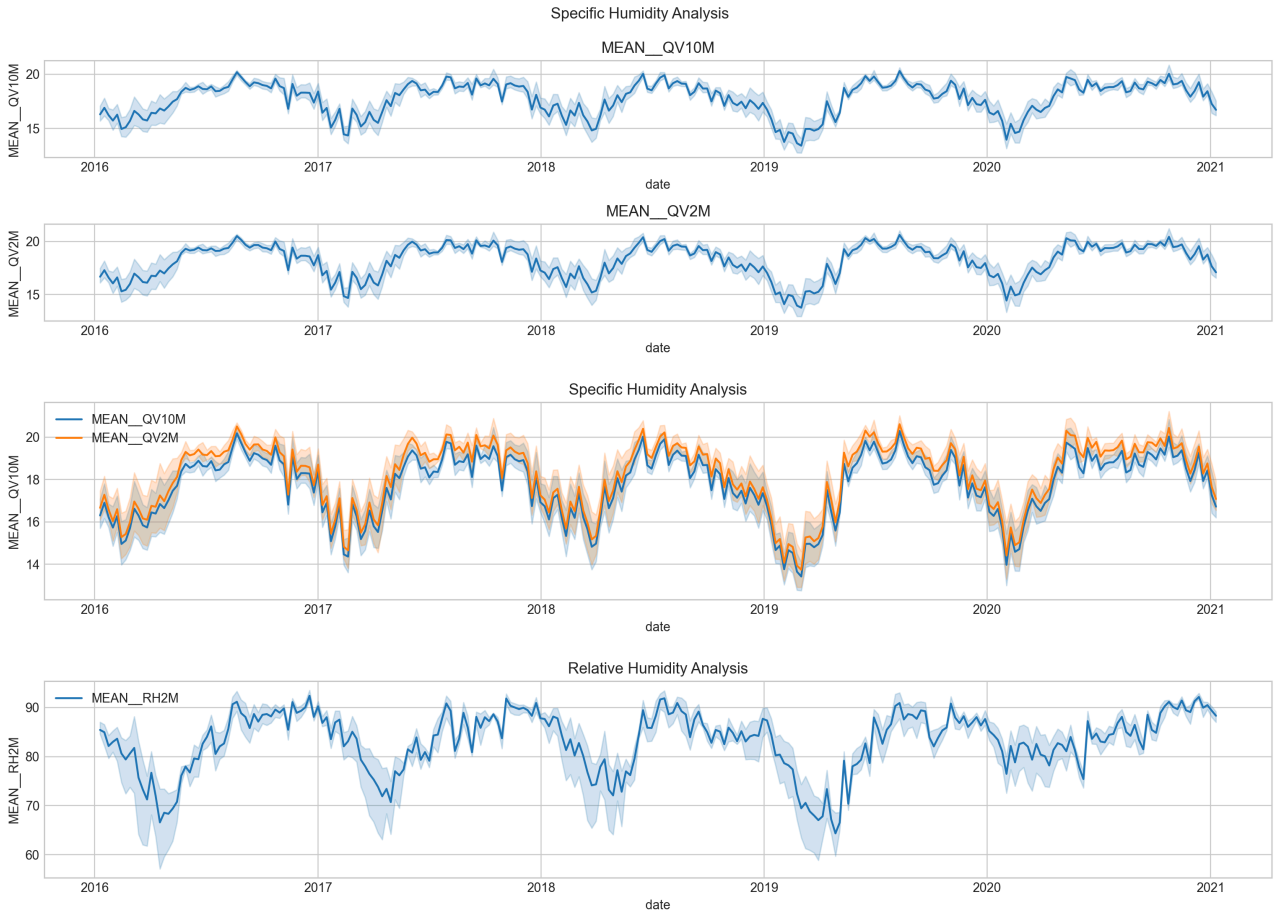


In [544]:
```
climate_plot(precip_, 'Precipitation Analysis')
climate_plot_combine(precip_, 'Precipitation Analysis')
```



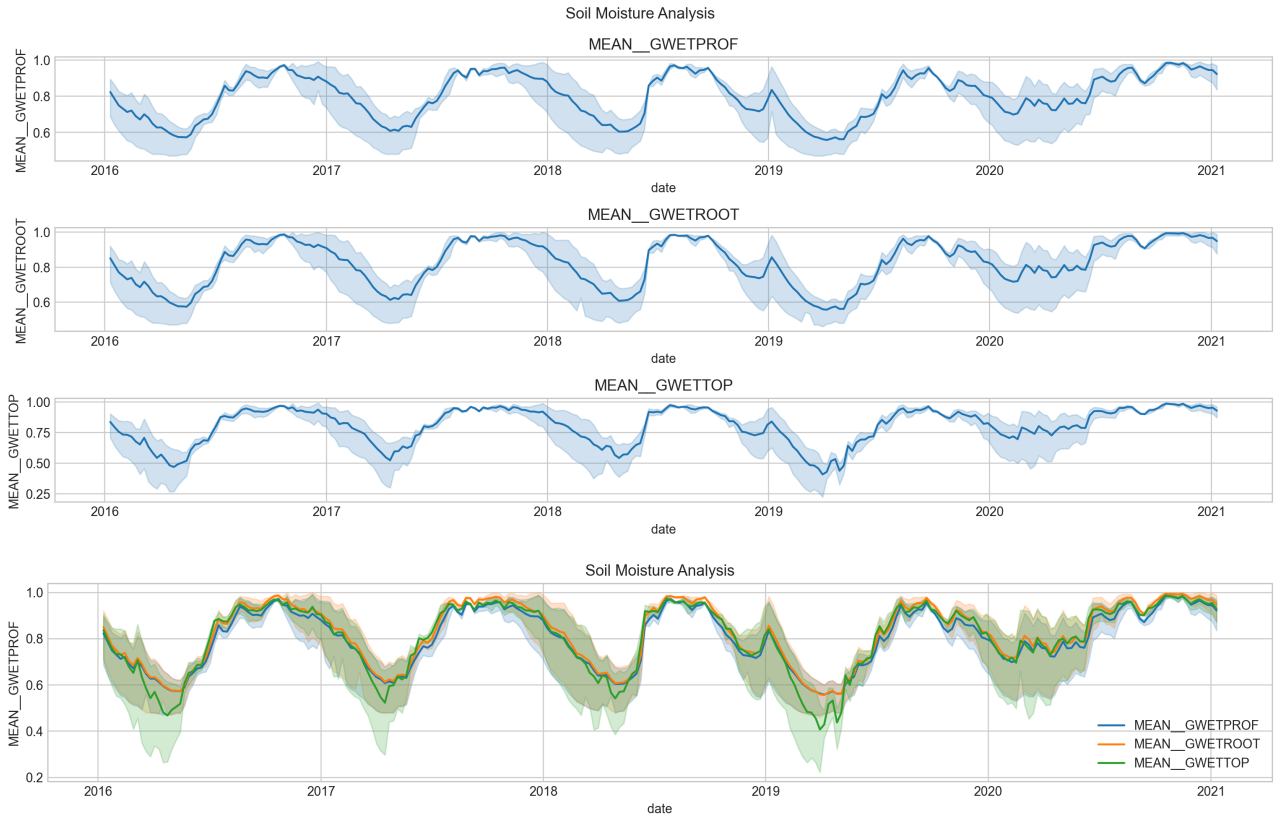
In [500]:
```
water_vapor = water_vapor_
precip = precip_
```

In [501]:
```
climate_plot(specific_humidity_, 'Specific Humidity Analysis')
climate_plot_combine(specific_humidity_, 'Specific Humidity Analysis')
climate_plot_combine(relative_humidity_, 'Relative Humidity Analysis')
```



In [502]:
```
specific_humidity = ['MEAN__QV2M']
relative_humidity = relative_humidity_
```

In [503]:
```
climate_plot(soil_moisture_, 'Soil Moisture Analysis')
climate_plot_combine(soil_moisture_, 'Soil Moisture Analysis')
```

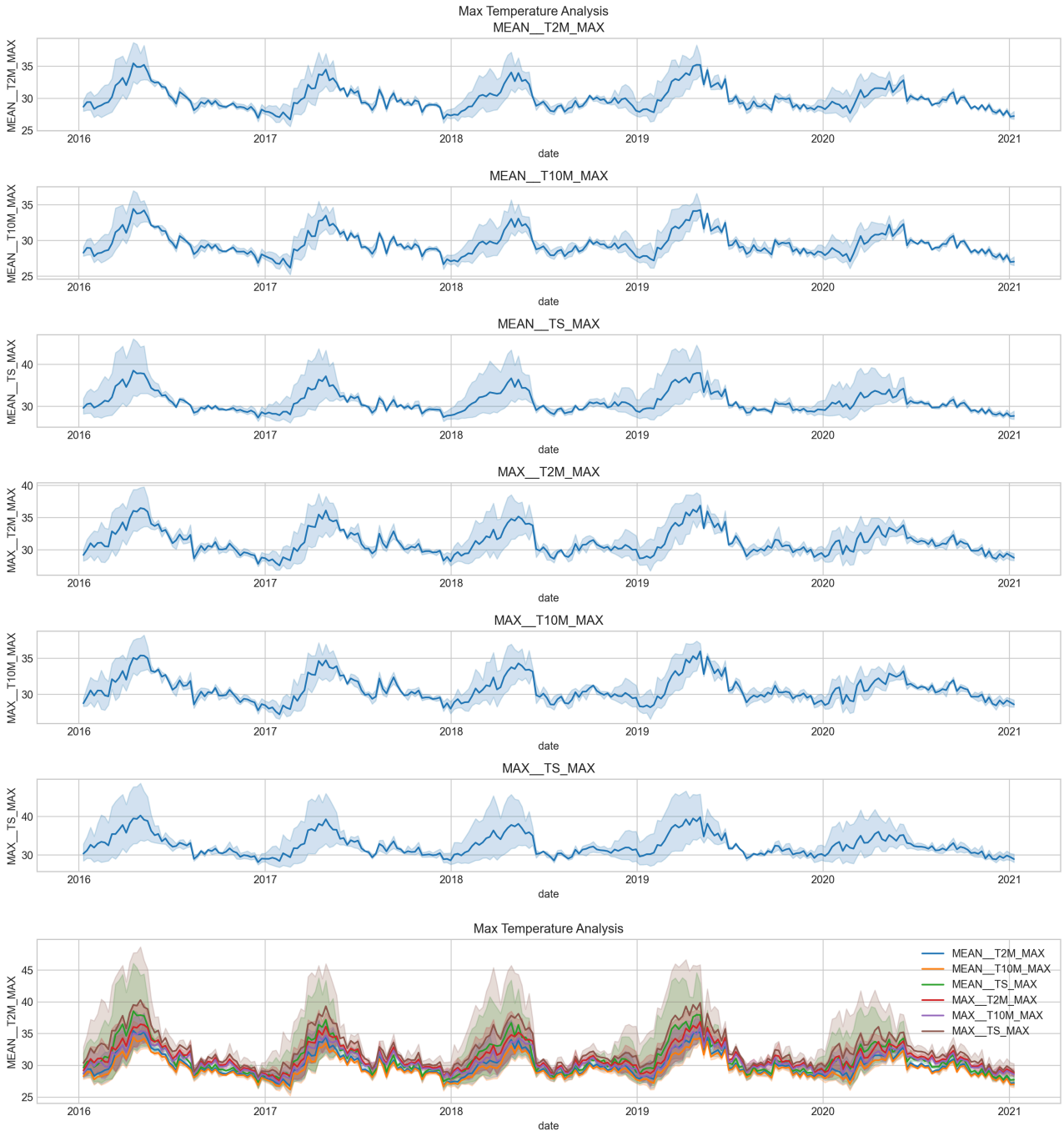


In [504]:
```
soil_moisture = soil_moisture_
```

In [505]:
```
climate_plot(temperature_mean_, 'Mean Temperature Analysis')
climate_plot_combine(temperature_mean_, 'Mean Temperature Analysis')
```

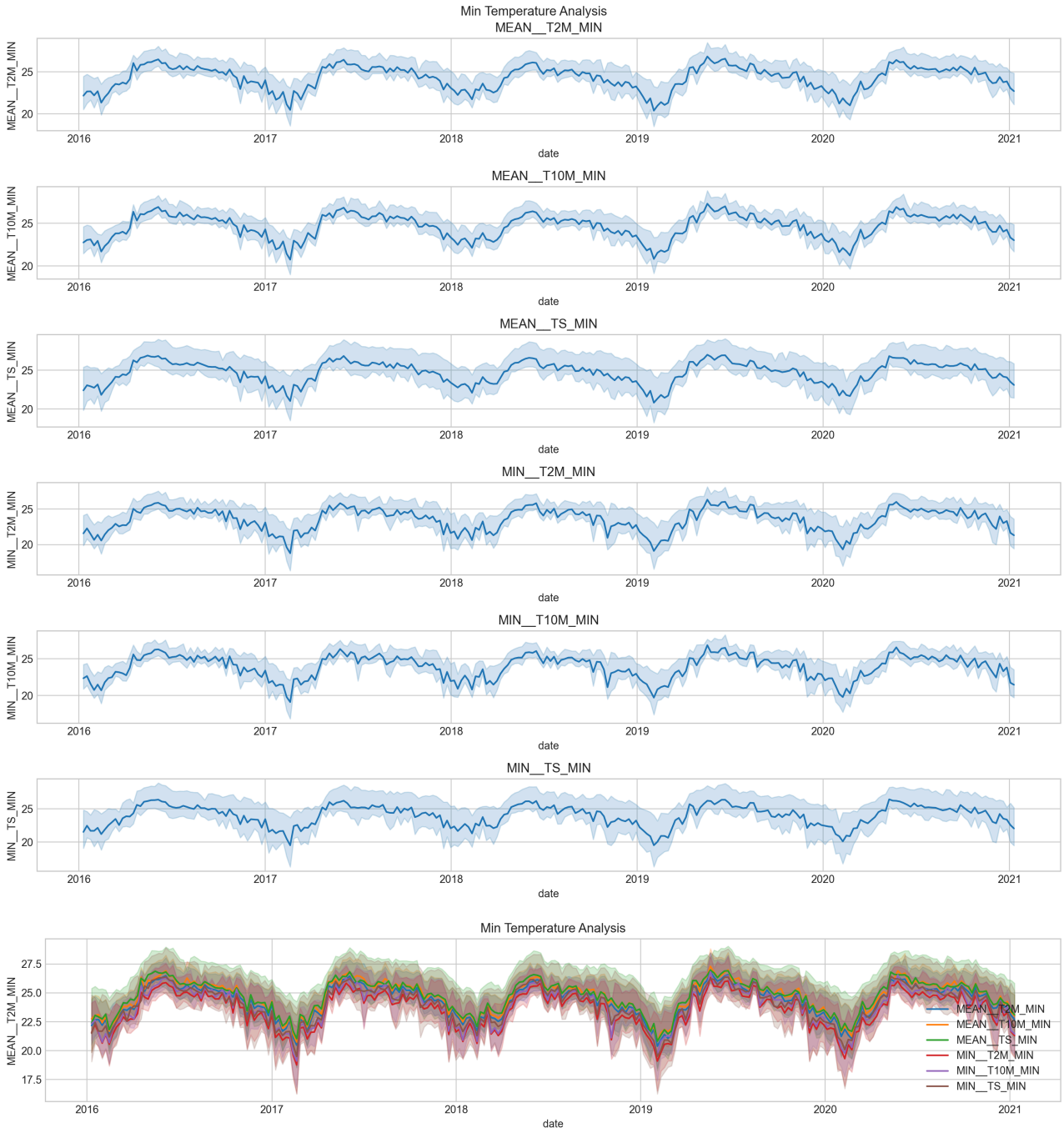
In [506]:
```
temperature_mean = temperature_mean_
```

In [507]:
```
climate_plot(temperature_max_, 'Max Temperature Analysis')
climate_plot_combine(temperature_max_, 'Max Temperature Analysis')
```



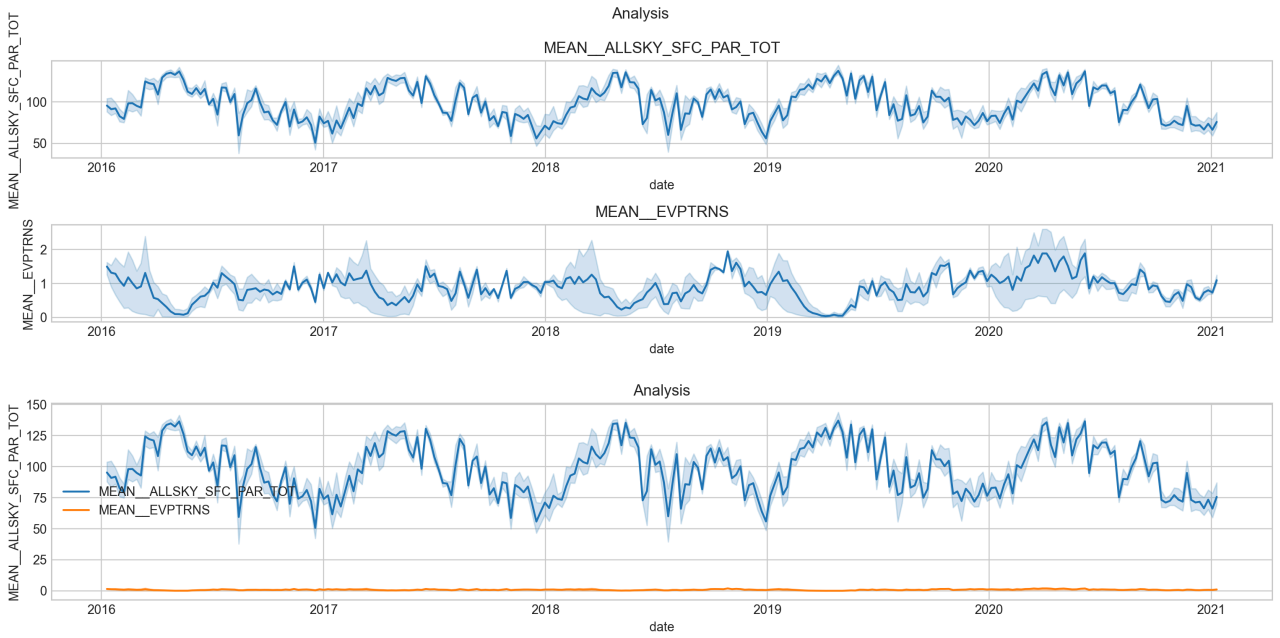
In [508]:
```
temperature_max = ['MAX__T2M_MAX', 'MAX__T10M_MAX', 'MAX__TS_MAX']
```

In [509]:
```
climate_plot(temperature_min_, 'Min Temperature Analysis')
climate_plot_combine(temperature_min_, 'Min Temperature Analysis')
```

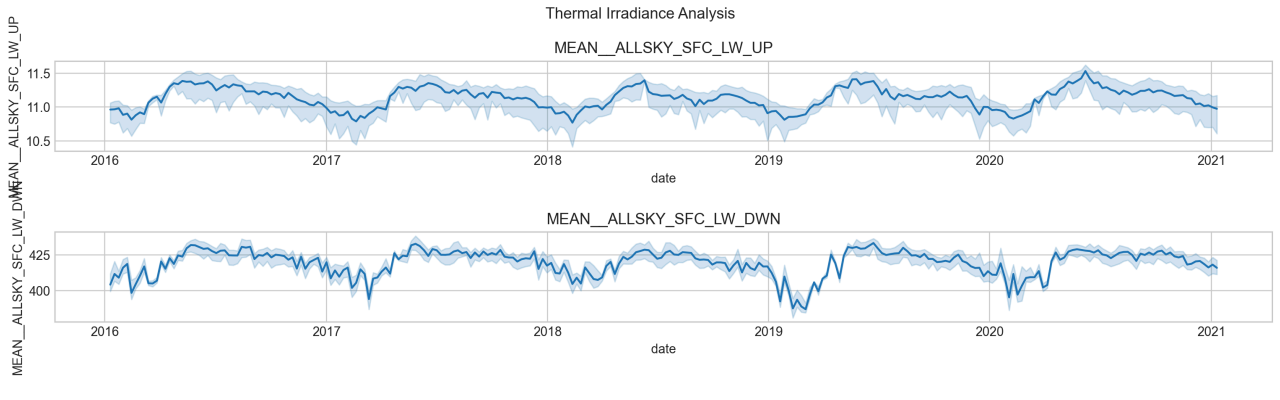


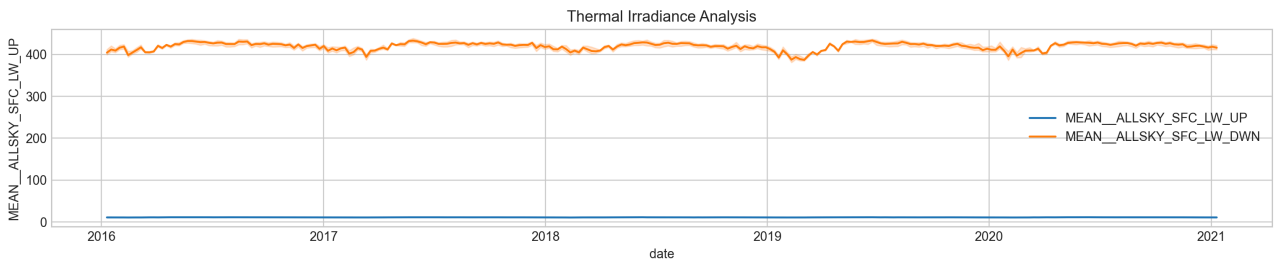
In [510]:
```
temperature_min = ['MIN__T2M_MIN', 'MIN__T10M_MIN', 'MIN__TS_MIN']
```

In [511]:
```
climate_plot(others, 'Analysis')
climate_plot_combine(others, 'Analysis')
```

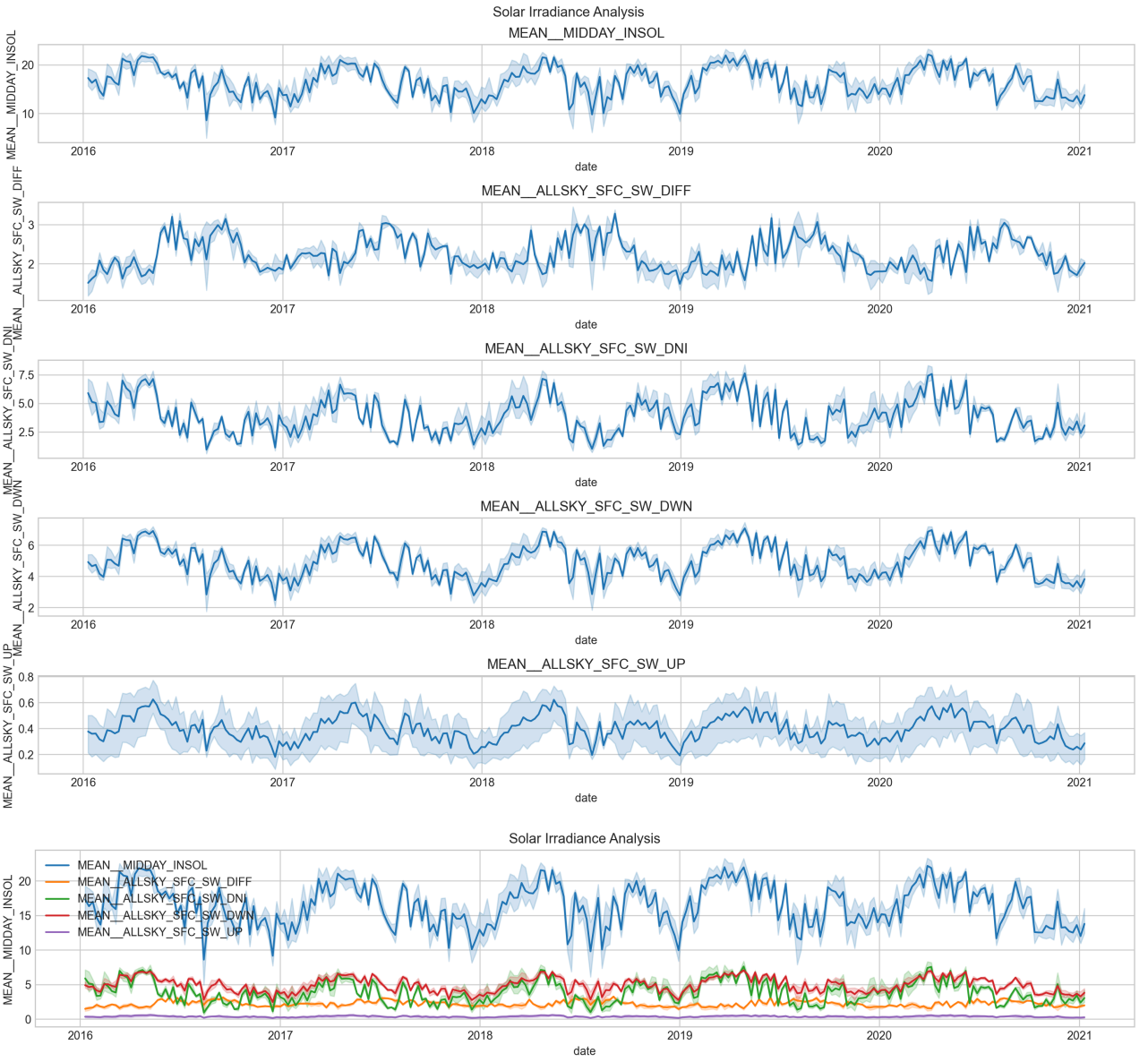


In [512]:
```
climate_plot(irradiance_lw, 'Thermal Irradiance Analysis')
climate_plot_combine(irradiance_lw, 'Thermal Irradiance Analysis')
```

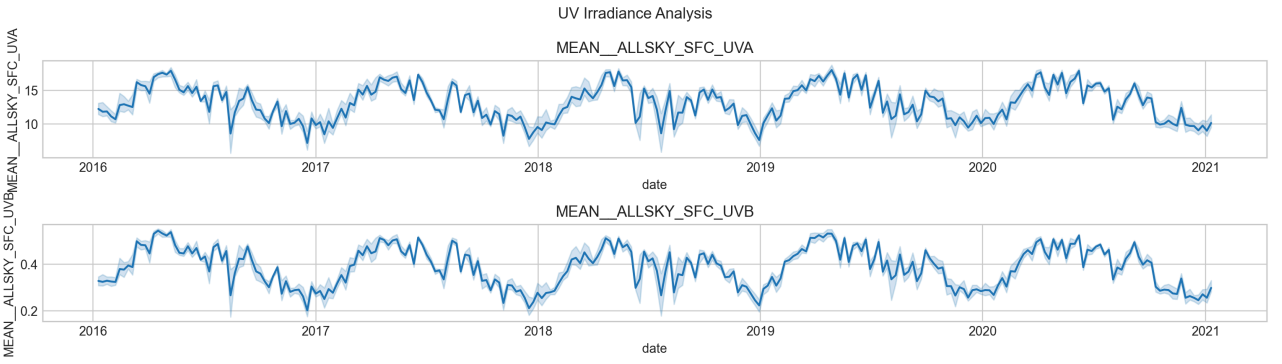


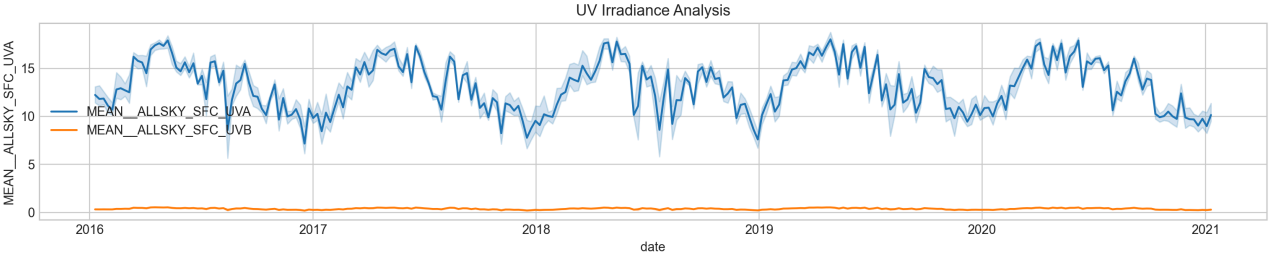


In [513]:
```
climate_plot(irradiance_sw, 'Solar Irradiance Analysis')
climate_plot_combine(irradiance_sw, 'Solar Irradiance Analysis')
```

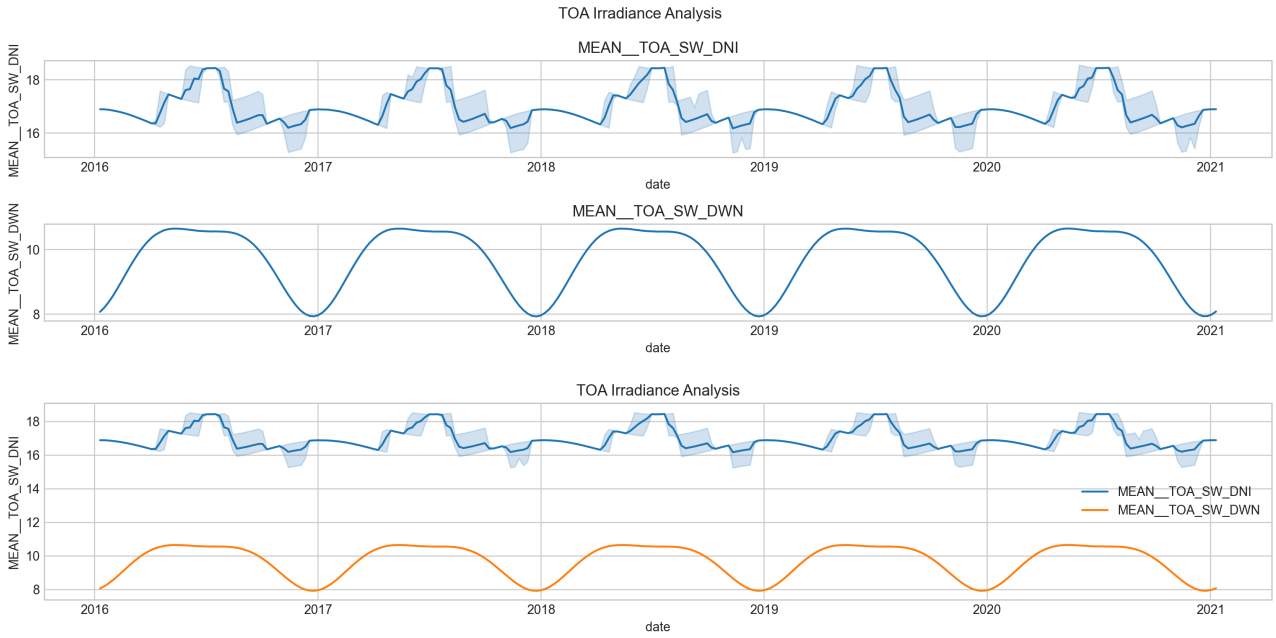


In [514]:
```
climate_plot(irradiance_uv, 'UV Irradiance Analysis')
climate_plot_combine(irradiance_uv, 'UV Irradiance Analysis')
```





In [515]:
```
climate_plot(irradiance_toa, 'TOA Irradiance Analysis')
climate_plot_combine(irradiance_toa, 'TOA Irradiance Analysis')
```



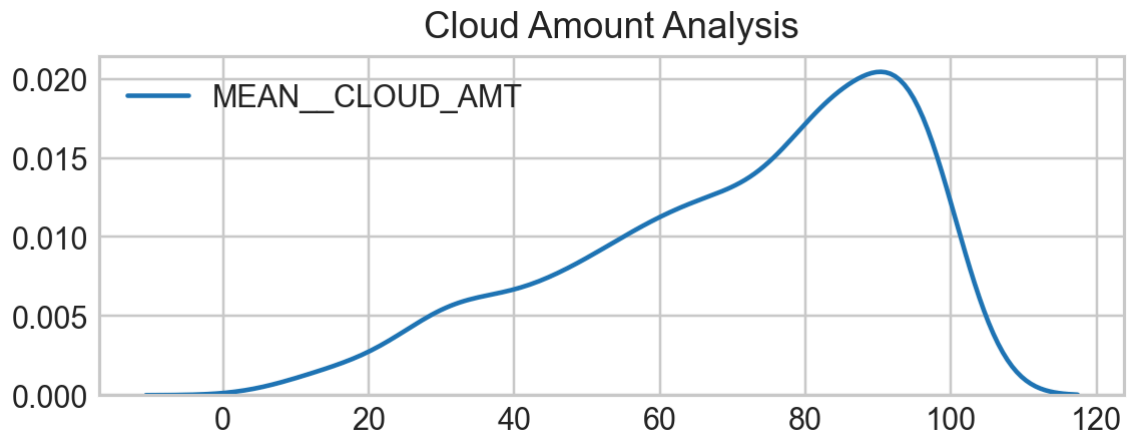
In [516]:
```
irradiance = ['MEAN__ALLSKY_SFC_LW_DWN', 'MEAN__ALLSKY_SFC_SW_DWN']
```

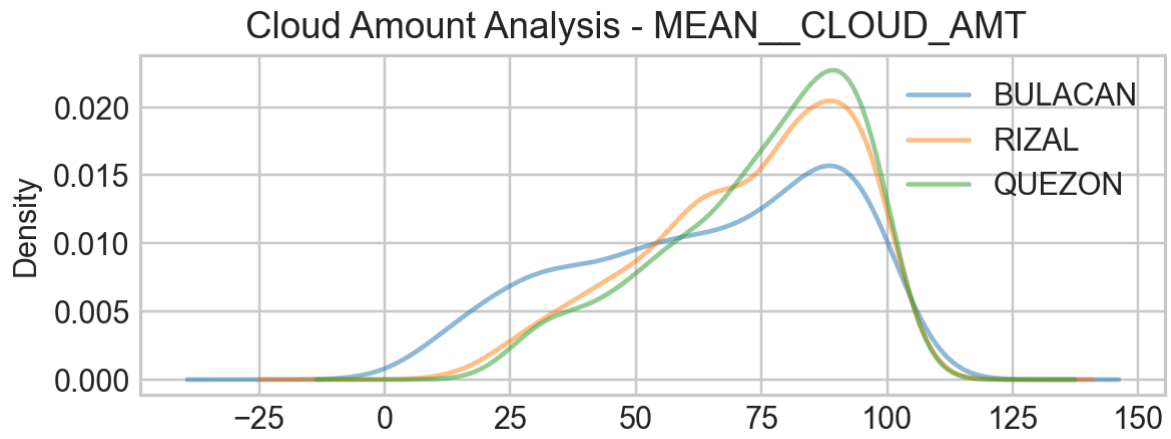
In [517]:
```
temperature = temperature_mean + temperature_min + temperature_max
climate = cloud_amt + water_vapor + precip + soil_moisture + temperature + irradiance + others
```

In [518]:
```
def kde(column, title):
    num_cols = 2
    num_rows = max(1, math.ceil(len(column) / num_cols))

    figsize = (6*num_cols, 2*num_rows)

    axes = df_climate[column].plot(
        kind='kde',
        subplots=True,
        figsize=figsize,
        layout=(num_rows, num_cols),
        sharex=False
    )
    fig = axes[0, 0].get_figure()
    fig.suptitle(title)
    plt.show()
```

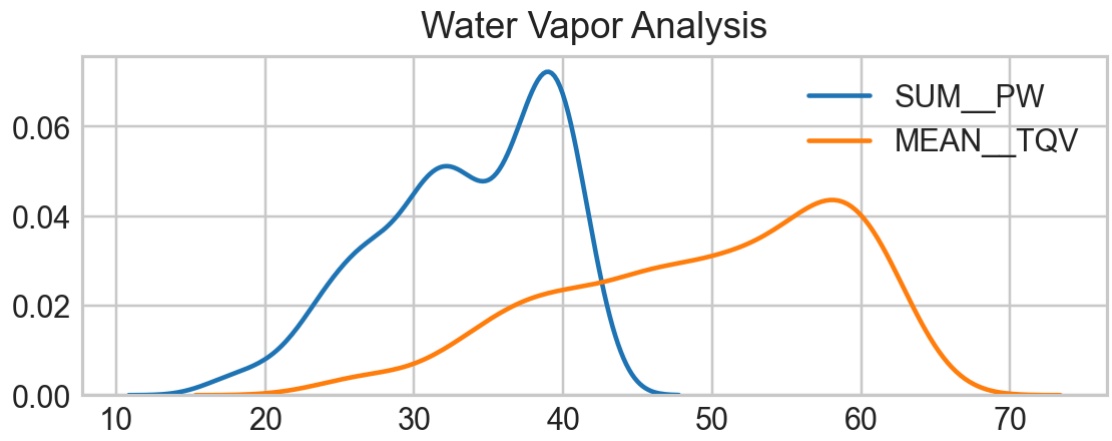
In [519]:
```
def kde(column, title):
    unique_locations = df_climate['loc'].unique()

    for col in column:
        fig, ax = plt.subplots(figsize=(6, 2))
        for location in unique_locations:
            #aa = df_climate[df_climate['loc'] == location][col]
            #print(location)
            #display(aa.head())
            df_climate[df_climate['loc'] == location][col].plot(kind='kde', ax=ax, label=location, alpha=0.5)
        ax.legend()
        fig.suptitle(f"{title} - {col}")
        plt.show()

def kde_combine(column, title):
    plt.figure(figsize=(6, 2))

    for col in column:
        sns.kdeplot(df_climate[col], label=col)
        plt.gca().set_ylabel('')

    plt.title(title)
    plt.legend()
    plt.gca().set_xlabel('')
    plt.show()
```

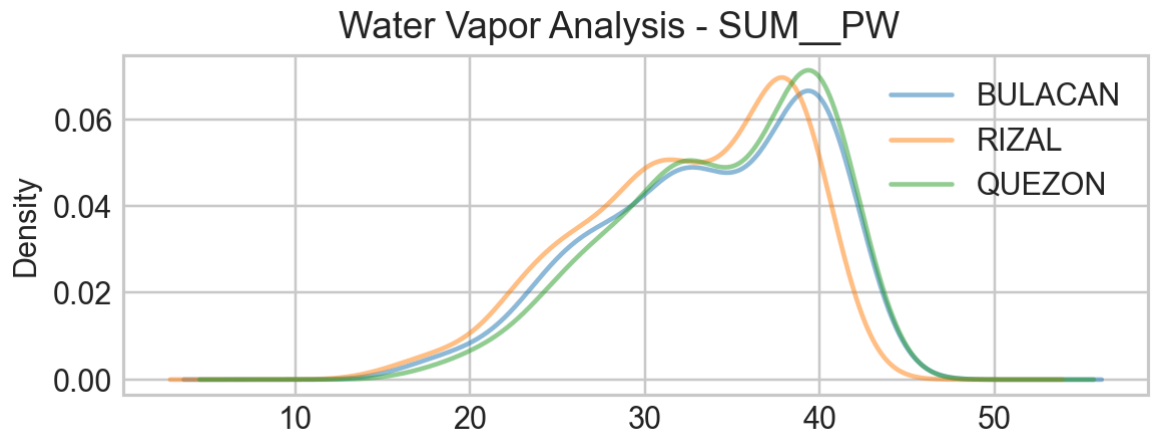
In [520]:
```
kde_combine(cloud_amt, 'Cloud Amount Analysis')
kde(cloud_amt, 'Cloud Amount Analysis')
```

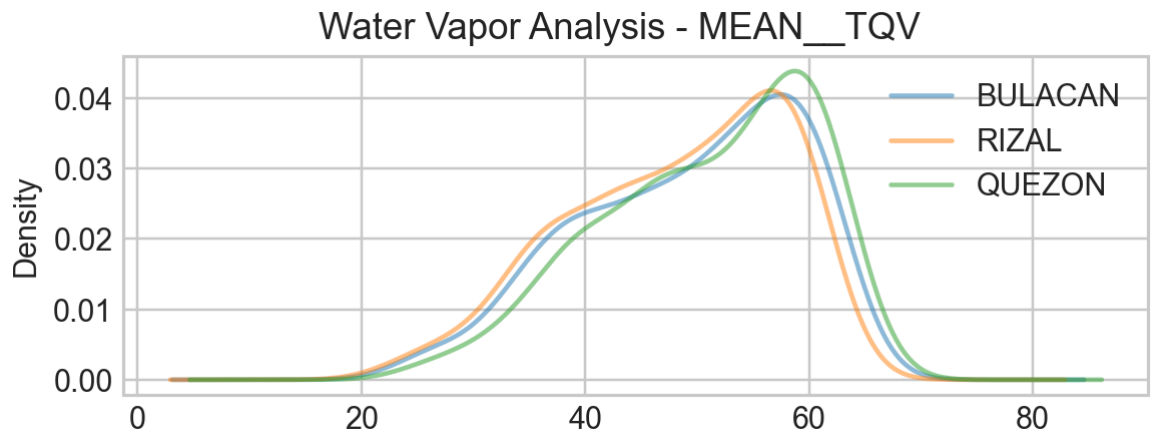




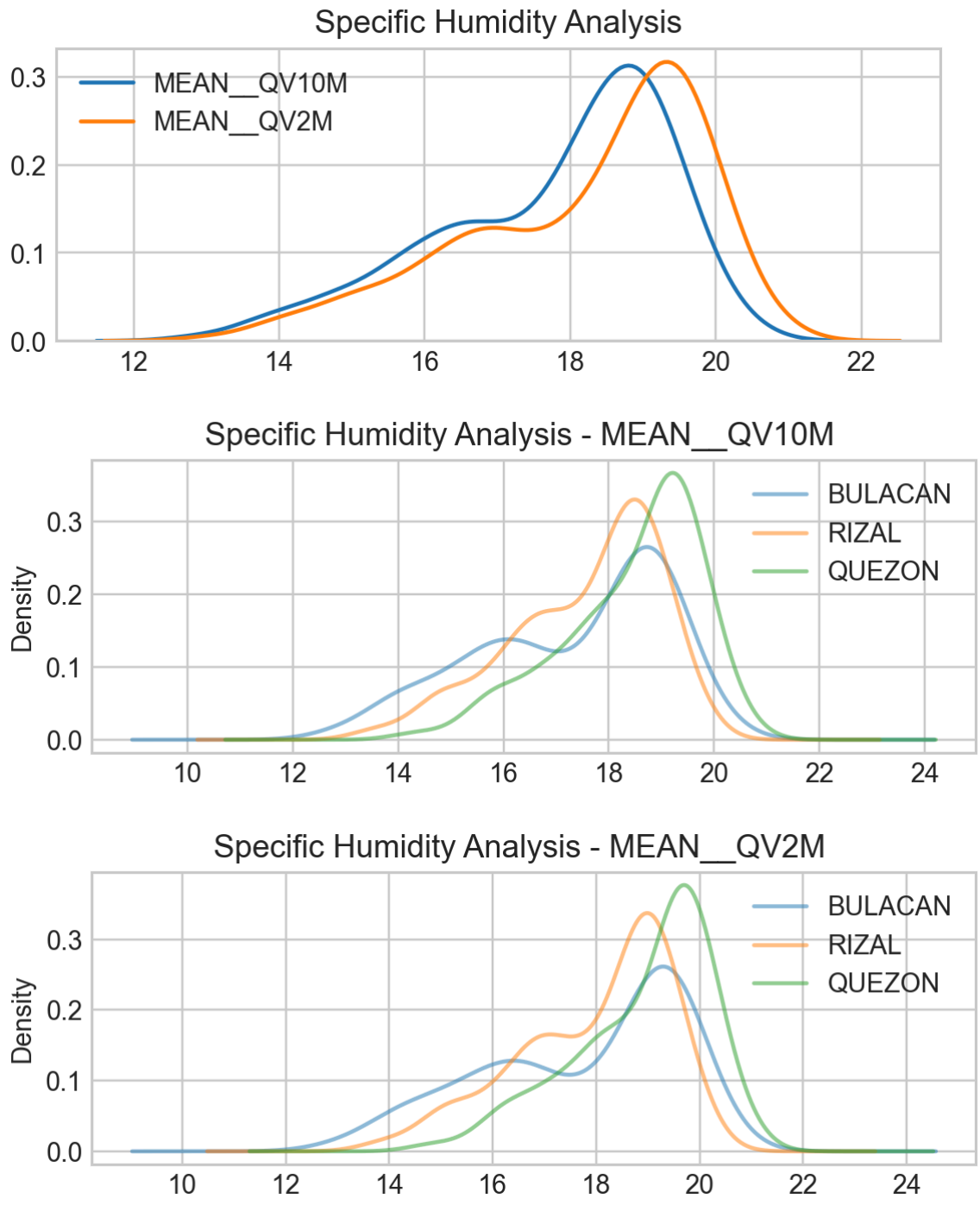
In [521]:
```
kde_combine(water_vapor, 'Water Vapor Analysis')
kde(water_vapor, 'Water Vapor Analysis')
```



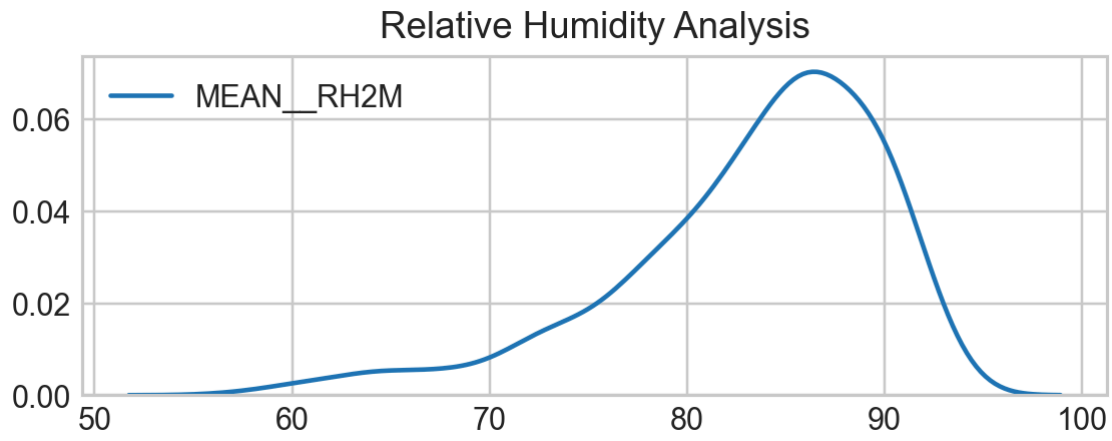


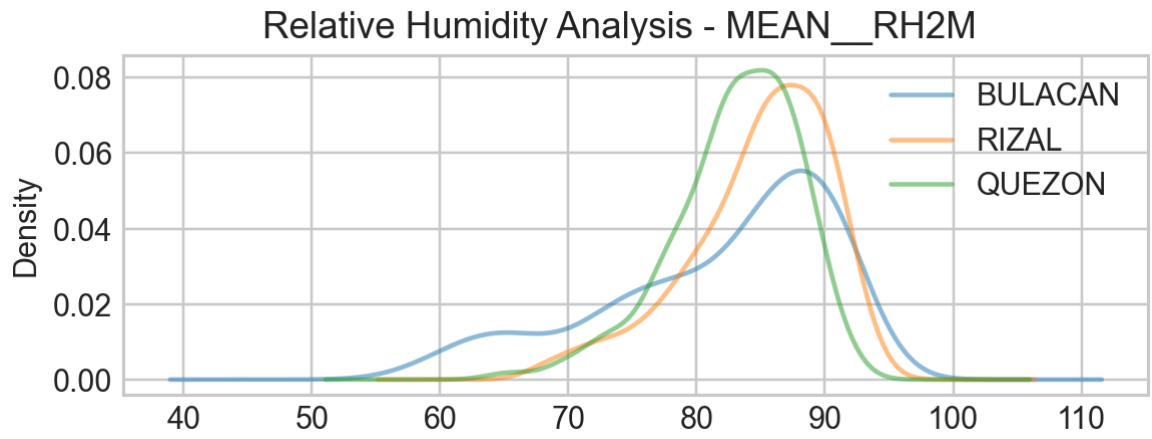


In [522]:
```
kde_combine(specific_humidity, 'Specific Humidity Analysis')
kde(specific_humidity, 'Specific Humidity Analysis')
```

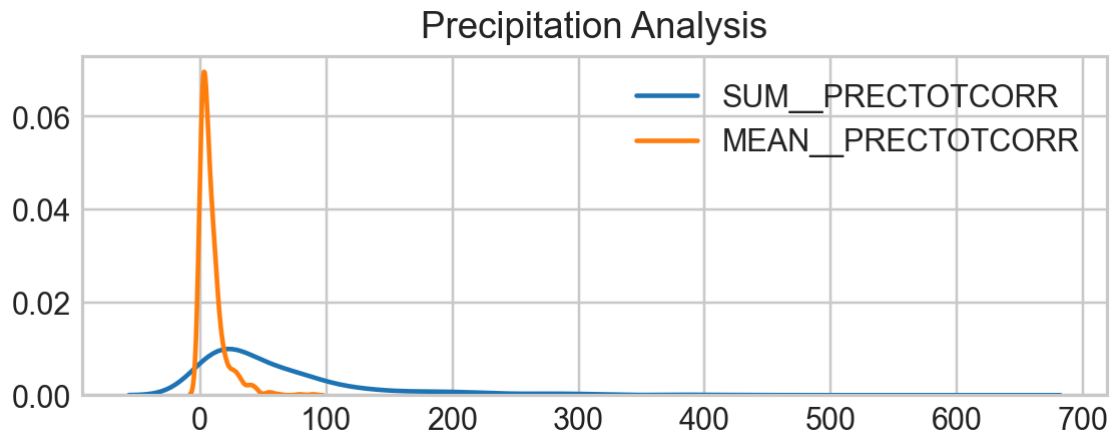


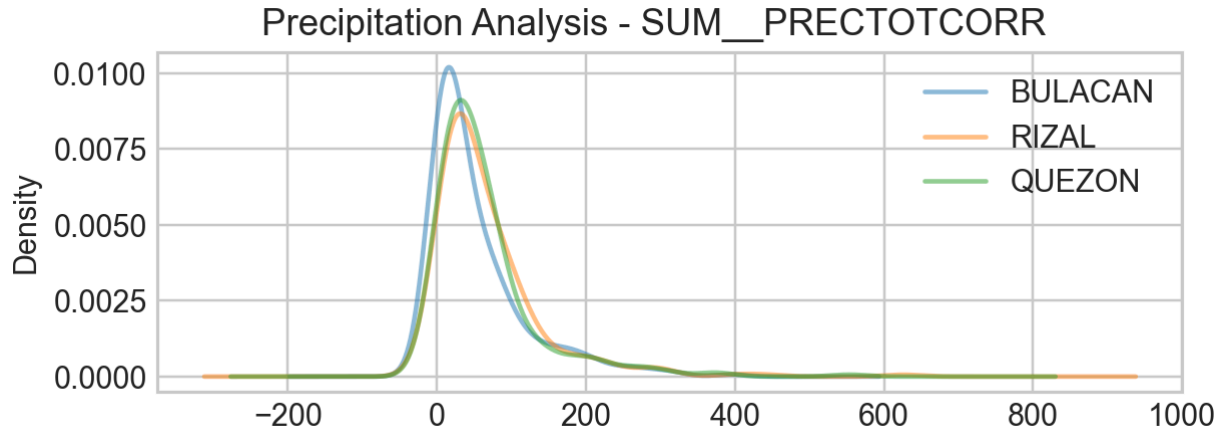
In [523]:
```
kde_combine(relative_humidity, 'Relative Humidity Analysis')
kde(relative_humidity, 'Relative Humidity Analysis')
```

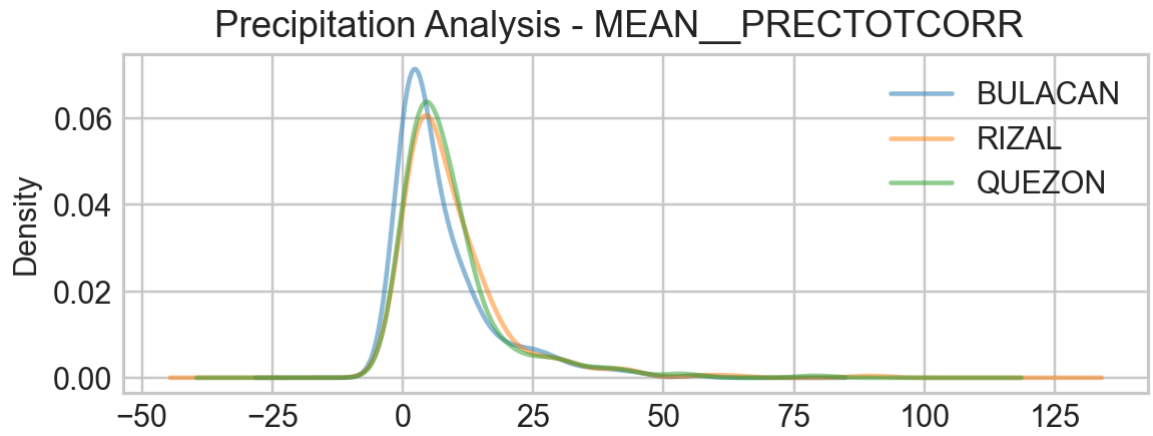




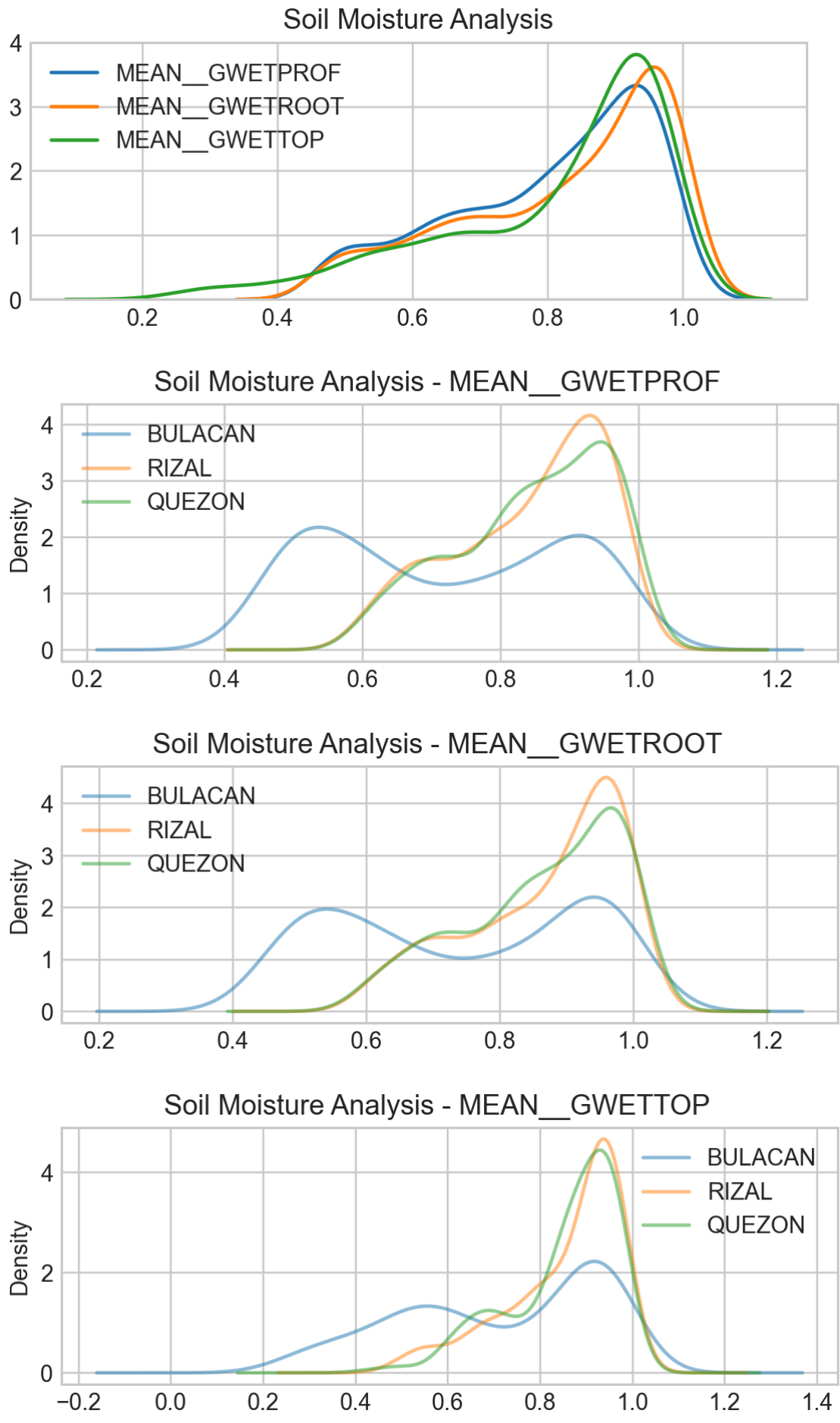
In [524]:
```
kde_combine(precip, 'Precipitation Analysis')
kde(precip, 'Precipitation Analysis')
```



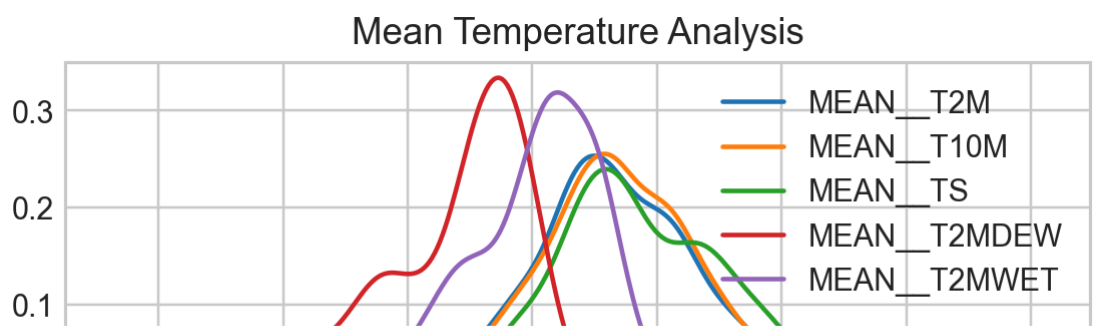




In [525]:
```
kde_combine(soil_moisture, 'Soil Moisture Analysis')
kde(soil_moisture, 'Soil Moisture Analysis')
```

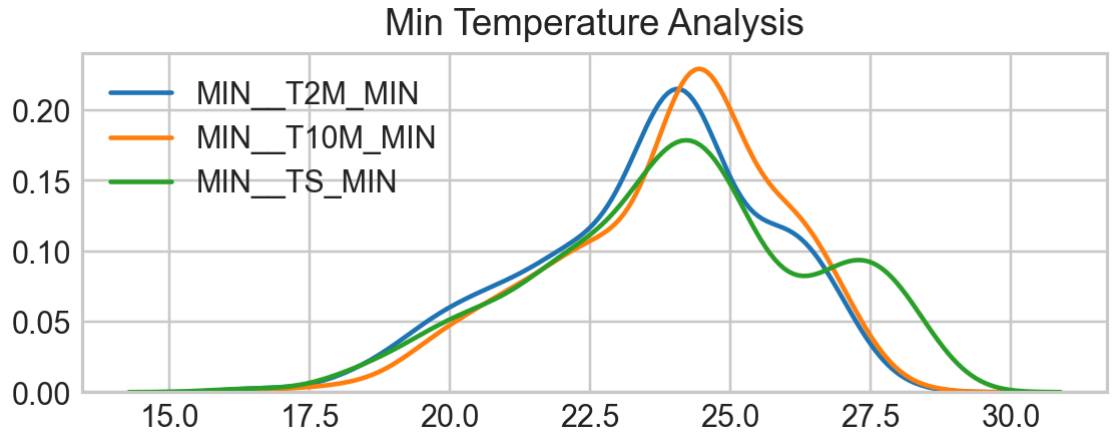


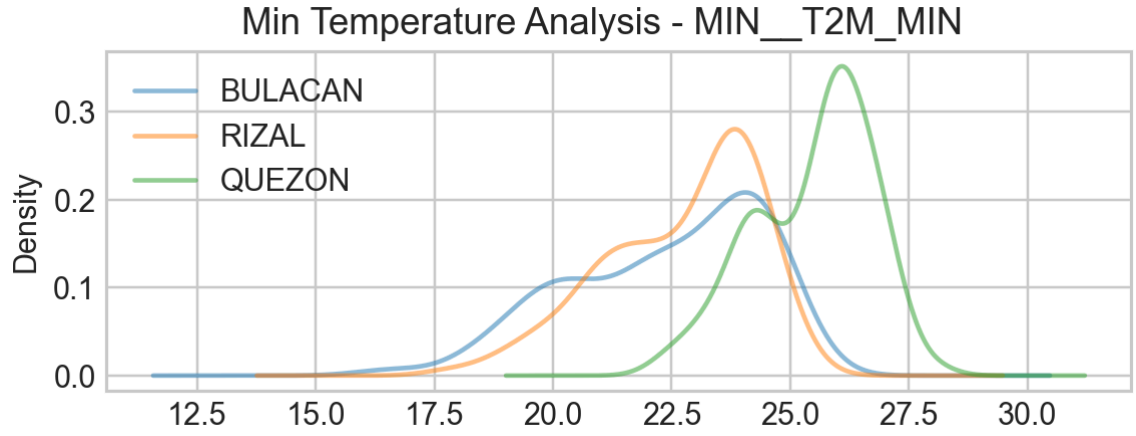
In [526]:
```
kde_combine(temperature_mean, 'Mean Temperature Analysis')
kde(temperature_mean, 'Mean Temperature Analysis')
```

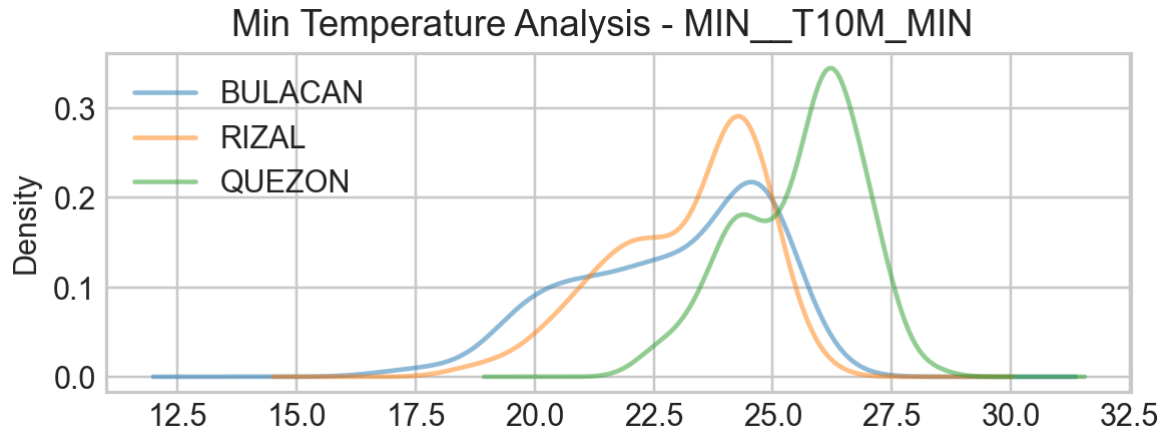


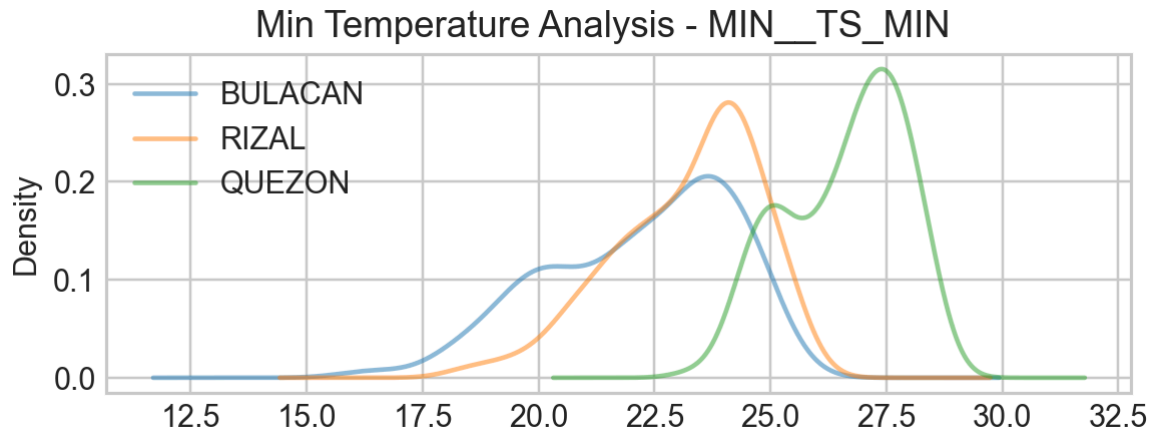
In [527]:

```
kde_combine(temperature_min, 'Min Temperature Analysis')
kde(temperature_min, 'Min Temperature Analysis')
```



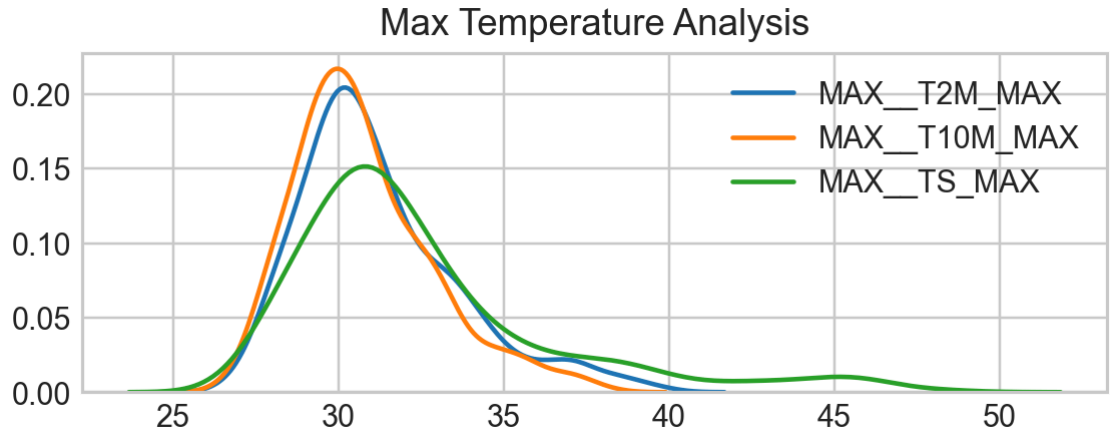


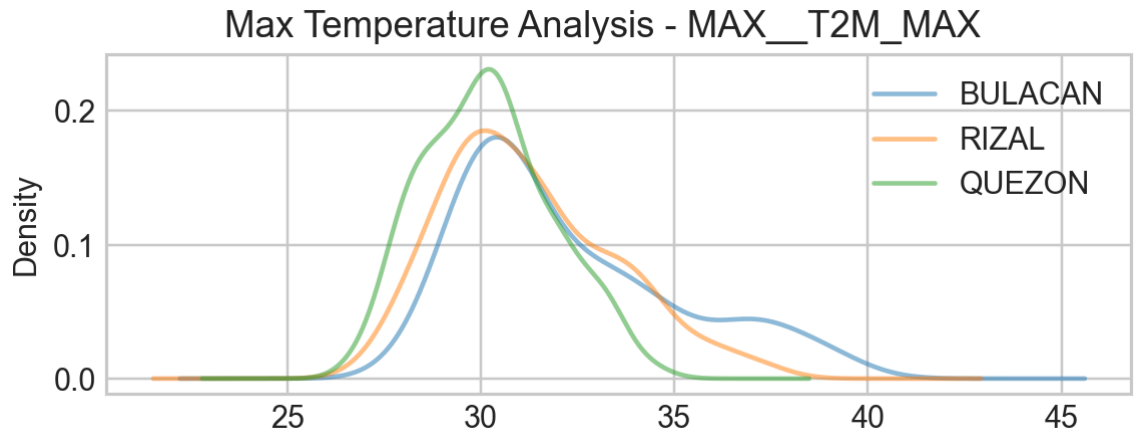


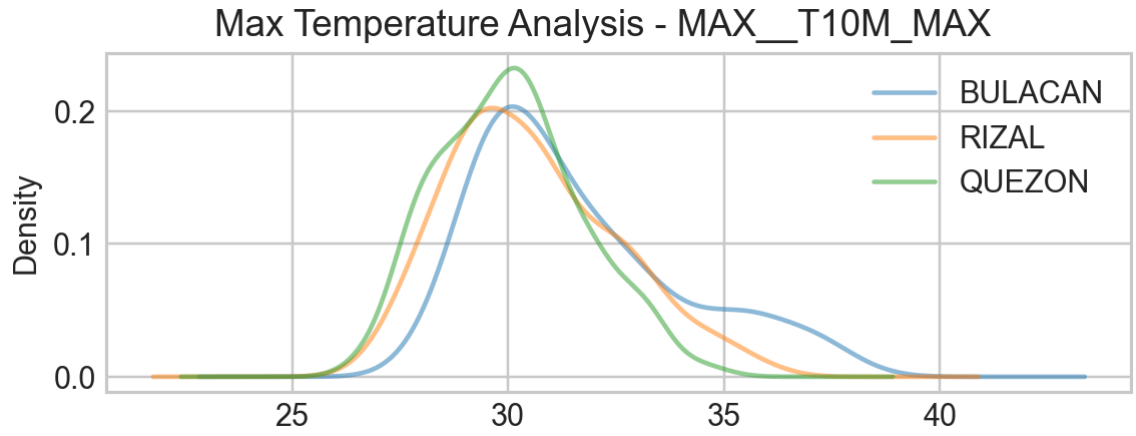


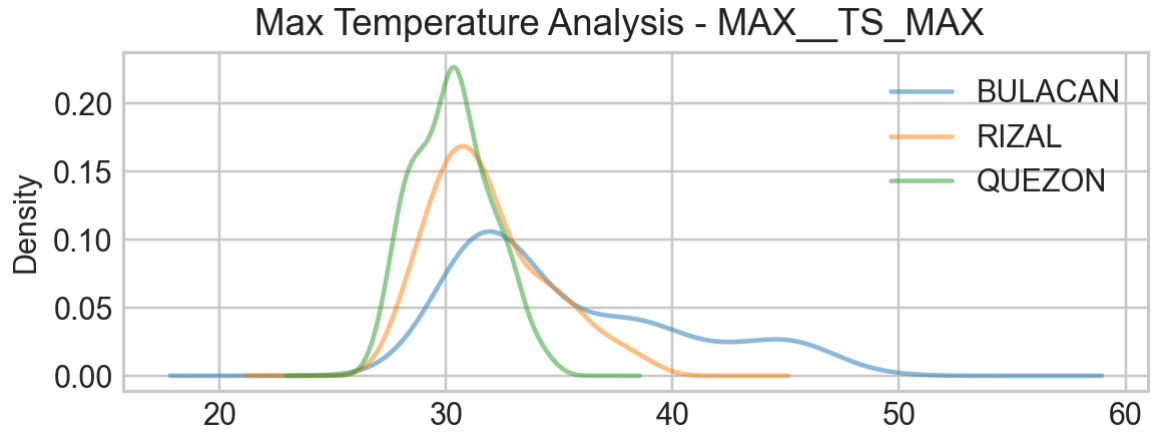
In [528]:

```
kde_combine(temperature_max, 'Max Temperature Analysis')
kde(temperature_max, 'Max Temperature Analysis')
```

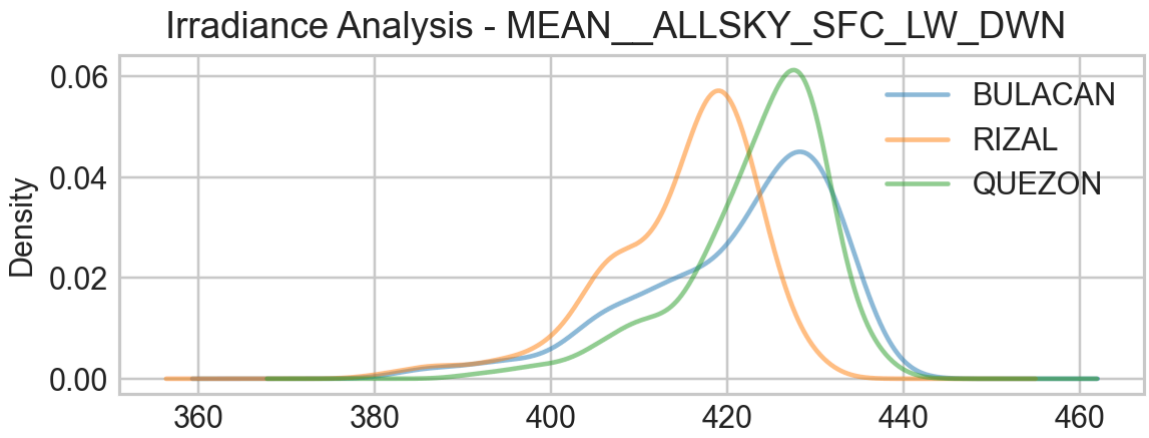


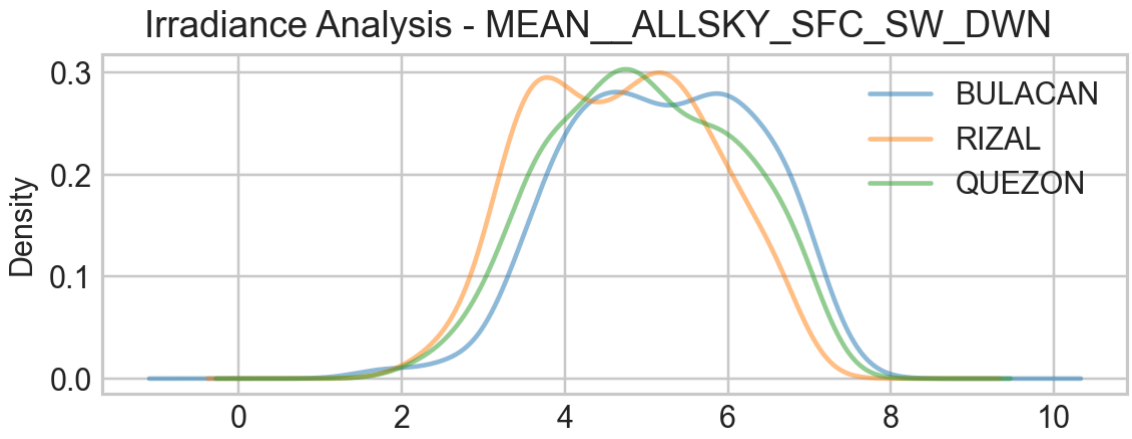


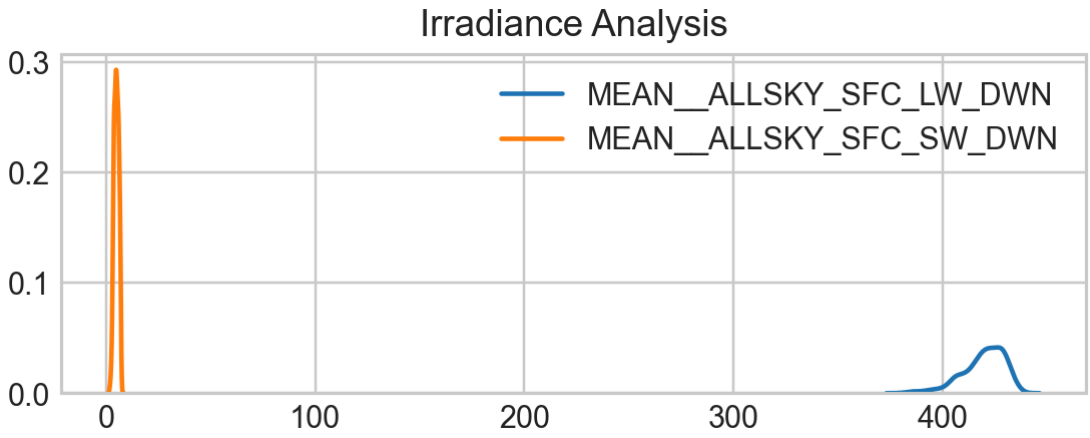




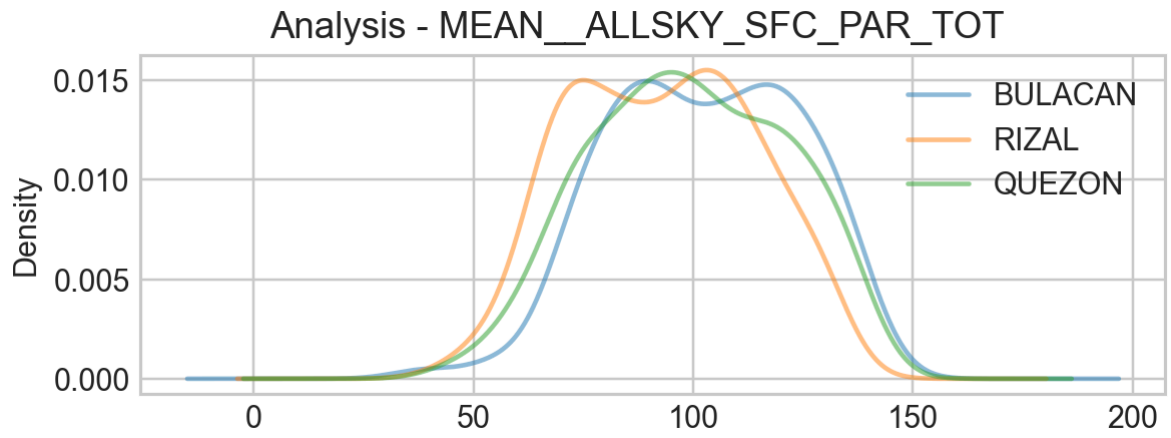
In [529]:
```
kde(irradiance, 'Irradiance Analysis')
kde_combine(irradiance, 'Irradiance Analysis')
```

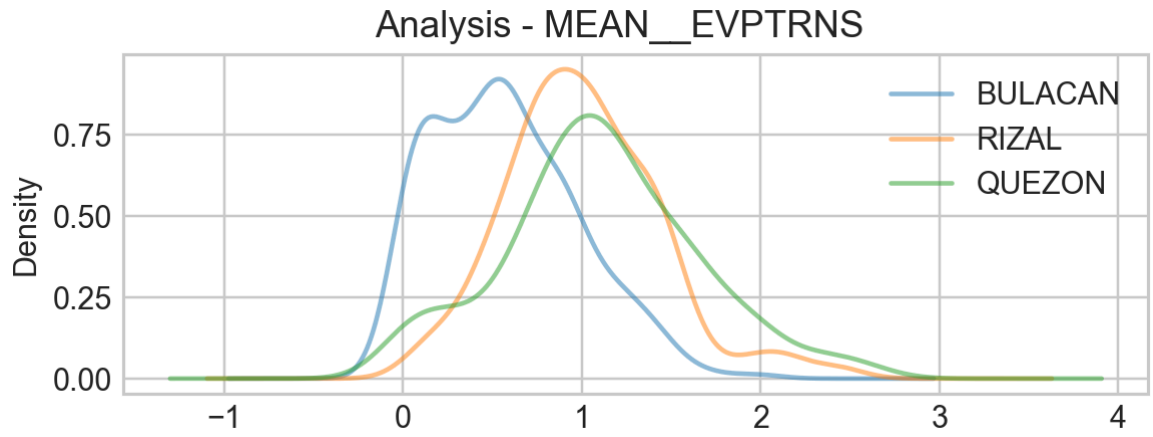


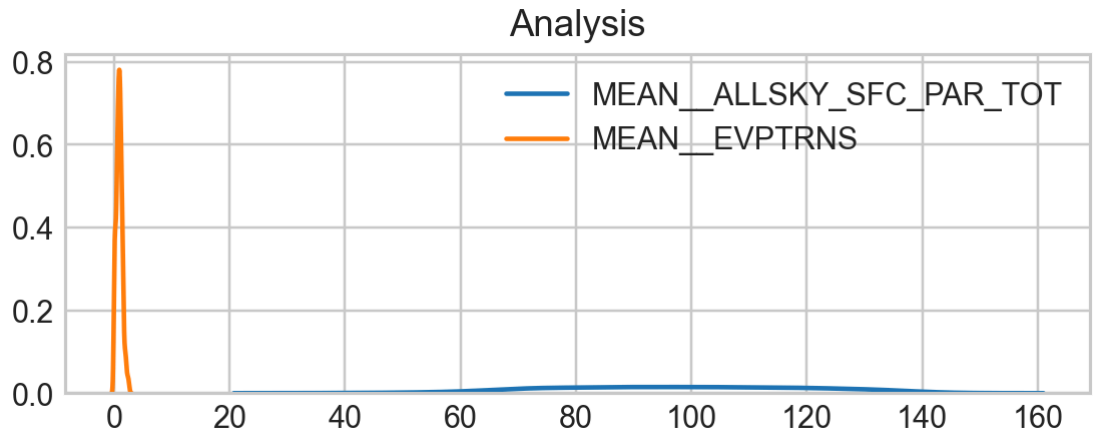




In [530]:
```
kde(others, 'Analysis')
kde_combine(others, 'Analysis')
```







In [531]:
```
ph_climate = ['loc'] + climate
df_climate_ph = df_climate[ph_climate]

print(f"Columns: {np.array(df_climate_ph.columns.tolist())}")
display(df_climate_ph.head())
```

```
Columns: ['loc' 'MEAN__CLOUD_AMT' 'SUM__PW' 'MEAN__TQV' 'SUM__PRECTOTCORR'
 'MEAN__PRECTOTCORR' 'MEAN__GWETPROF' 'MEAN__GWETROOT' 'MEAN__GWETTOP'
 'MEAN__T2M' 'MEAN__T10M' 'MEAN__TS' 'MEAN__T2MDEW' 'MEAN__T2MWET'
 'MIN__T2M_MIN' 'MIN__T10M_MIN' 'MIN__TS_MIN' 'MAX__T2M_MAX'
 'MAX__T10M_MAX' 'MAX__TS_MAX' 'MEAN__ALLSKY_SFC_LW_DWN'
 'MEAN__ALLSKY_SFC_SW_DWN' 'MEAN__ALLSKY_SFC_PAR_TOT' 'MEAN__EVPTRNS']
```

date	loc	MEAN__CLOUD_AMT	SUM__PW	MEAN__TQV	SUM__PRECTOTCORR	MEAN__PRECTOTCORR	MEAN__GWETPROF	MEAN__GWETROOT	MEAN__GWETTOP	MEAN__T2M	...	MIN__T2M_MIN
2016-01-10	BULACAN	21.675714	23.82	31.891429	0.64	0.091429	0.685714	0.711429	0.710000	24.465714	...	19.85
2016-01-17	BULACAN	32.798571	28.23	42.594286	0.47	0.067143	0.642857	0.664286	0.641429	25.495714	...	20.80
2016-01-24	BULACAN	24.114286	26.87	39.175714	2.54	0.362857	0.608571	0.622857	0.594286	25.640000	...	20.26
2016-01-31	BULACAN	54.452857	30.44	41.842857	1.77	0.252857	0.581429	0.591429	0.552857	24.665714	...	19.40
2016-02-07	BULACAN	58.200000	31.57	44.405714	14.96	2.137143	0.554286	0.564286	0.542857	25.388571	...	20.05

5 rows × 24 columns

In [532]:
```
df_all = df_dengue.merge(df_climate_ph, how='inner', on=['date', 'loc'])
df_all.head()
```

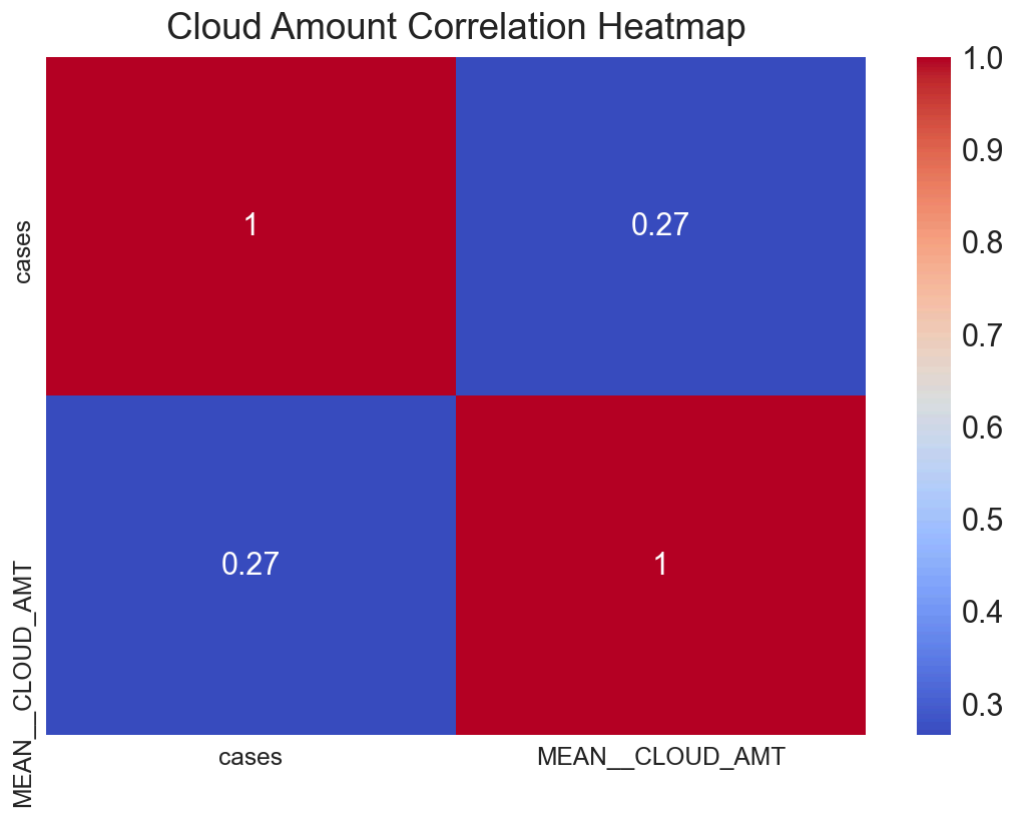
Out[532]:

date	loc	region	cases	deaths	year	month	MEAN__CLOUD_AMT	SUM__PW	MEAN__TQV	SUM__PRECTOTCORR	...	MIN__T2M_MIN	MIN__T10M_MIN	MIN__TS_MIN	MAX__T2M_MAX	MAX__
2016-01-10	BULACAN	REGION III-CENTRAL LUZON	97.0	0.0	2016	January	21.675714	23.82	31.891429	0.64	...	19.85	20.88	19.05	30.07	
2016-01-17	BULACAN	REGION III-CENTRAL LUZON	58.0	0.0	2016	January	32.798571	28.23	42.594286	0.47	...	20.80	21.37	20.65	31.60	
2016-01-24	BULACAN	REGION III-CENTRAL LUZON	76.0	1.0	2016	January	24.114286	26.87	39.175714	2.54	...	20.26	20.45	20.38	32.98	
2016-01-31	BULACAN	REGION III-CENTRAL LUZON	101.0	0.0	2016	January	54.452857	30.44	41.842857	1.77	...	19.40	19.63	19.30	32.95	
2016-02-07	BULACAN	REGION III-CENTRAL LUZON	85.0	0.0	2016	February	58.200000	31.57	44.405714	14.96	...	20.05	20.19	20.13	32.80	

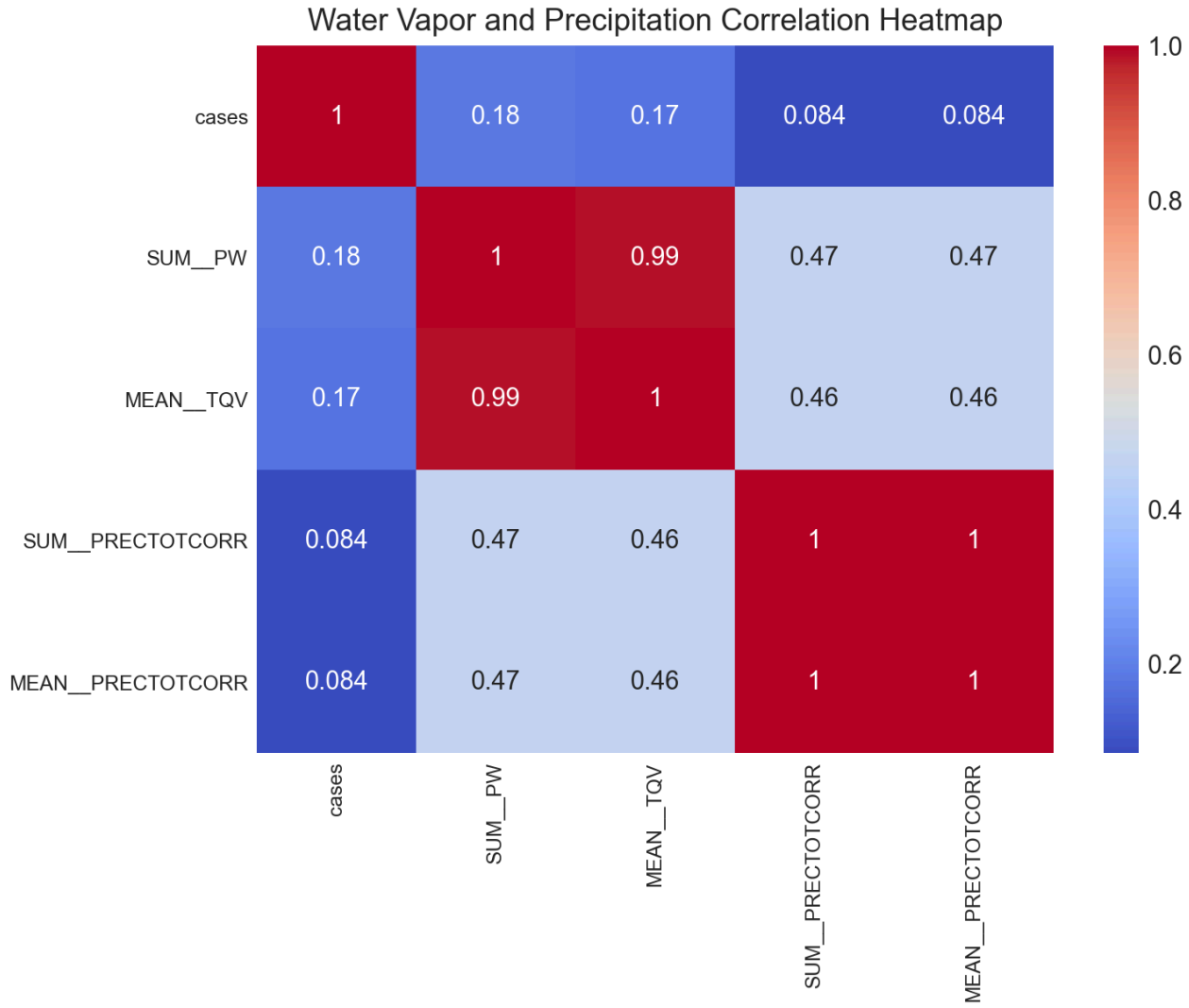
5 rows × 29 columns

In [533]:
```
temperature = temperature_mean + temperature_min + temperature_max
climate = cloud_amt + water_vapor + precip + soil_moisture + temperature + irradiance + others
```

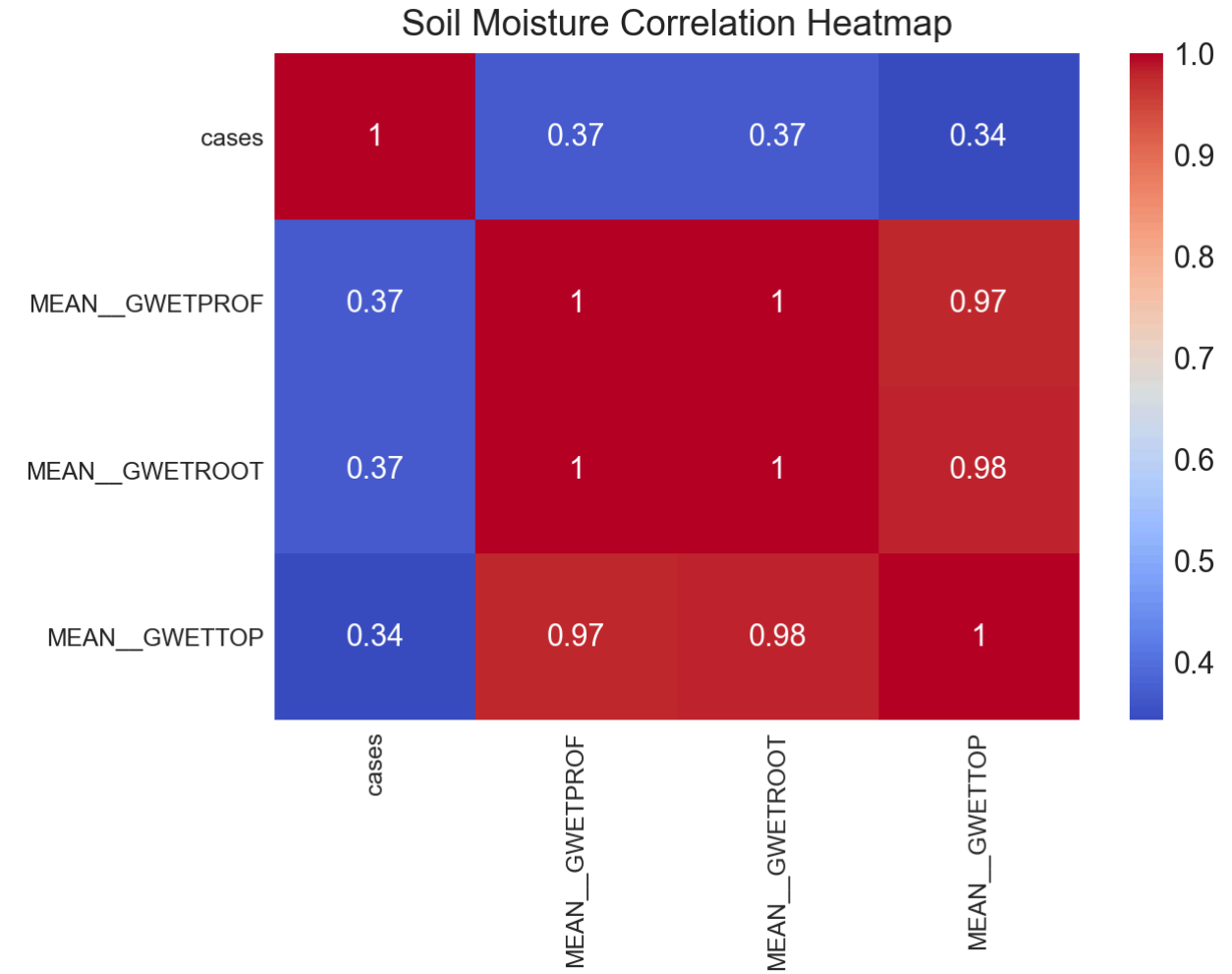
In [534]:
```
corr_col = ['cases'] + cloud_amt
df_corr = df_all[corr_col].corr()

cols = len(corr_col)
plt.figure(figsize=(2*cols+2, 2*cols))
ax = sns.heatmap(df_corr, annot=True, cmap='coolwarm')
ax.set_xticklabels(ax.get_xticklabels(), fontsize=8)  # Set x-axis label size
ax.set_yticklabels(ax.get_yticklabels(), fontsize=8)  # Set y-axis label size
plt.title('Cloud Amount Correlation Heatmap')
plt.show()
```



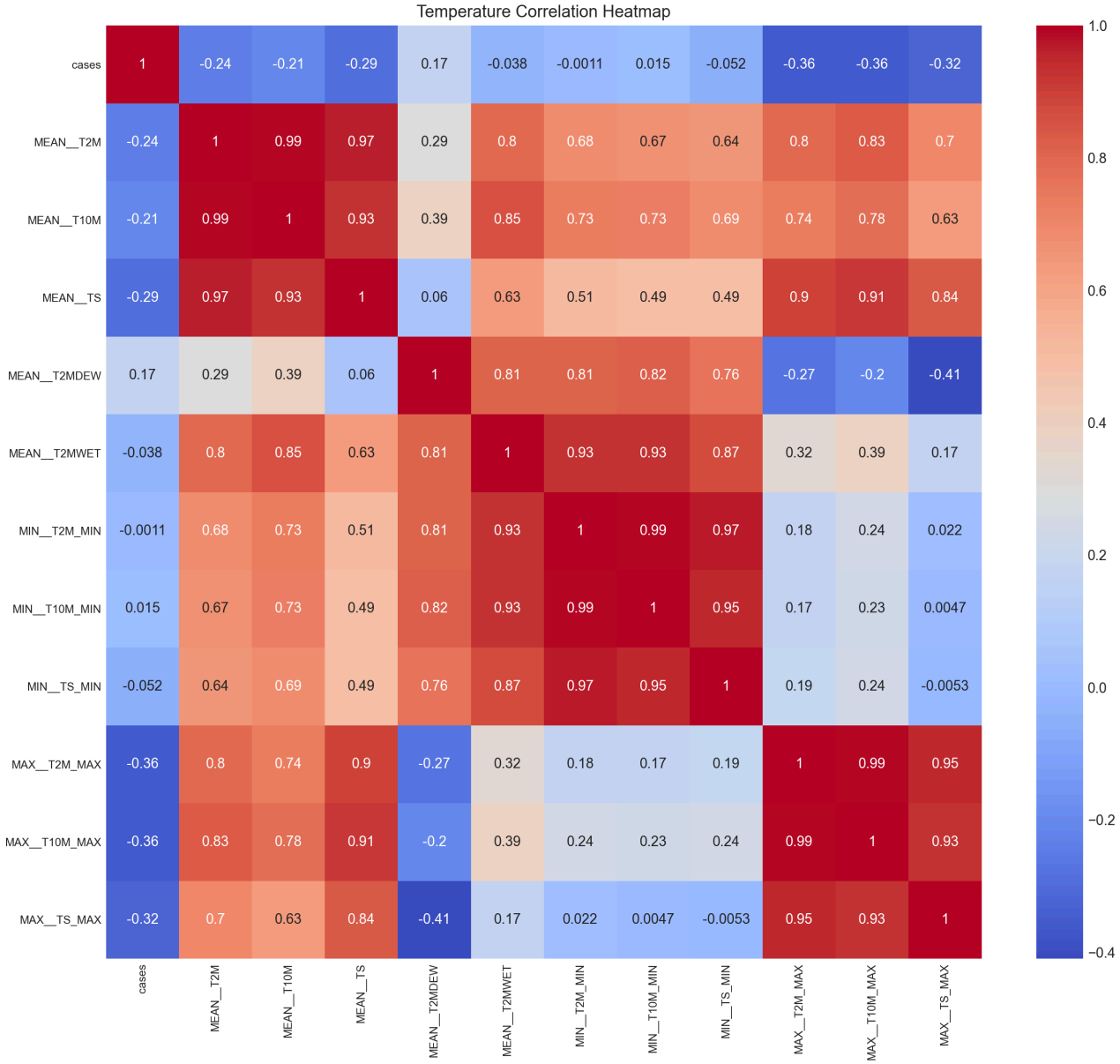
In [535]:
```
title = 'Water Vapor and Precipitation'
corr_col = ['cases'] + water_vapor + precip
df_corr = df_all[corr_col].corr()

cols = len(corr_col)
plt.figure(figsize=(cols+2, cols))
ax = sns.heatmap(df_corr, annot=True, cmap='coolwarm')
ax.set_xticklabels(ax.get_xticklabels(), fontsize=8)  # Set x-axis label size
ax.set_yticklabels(ax.get_yticklabels(), fontsize=8)  # Set y-axis label size
plt.title(title + ' Correlation Heatmap')
plt.show()
```



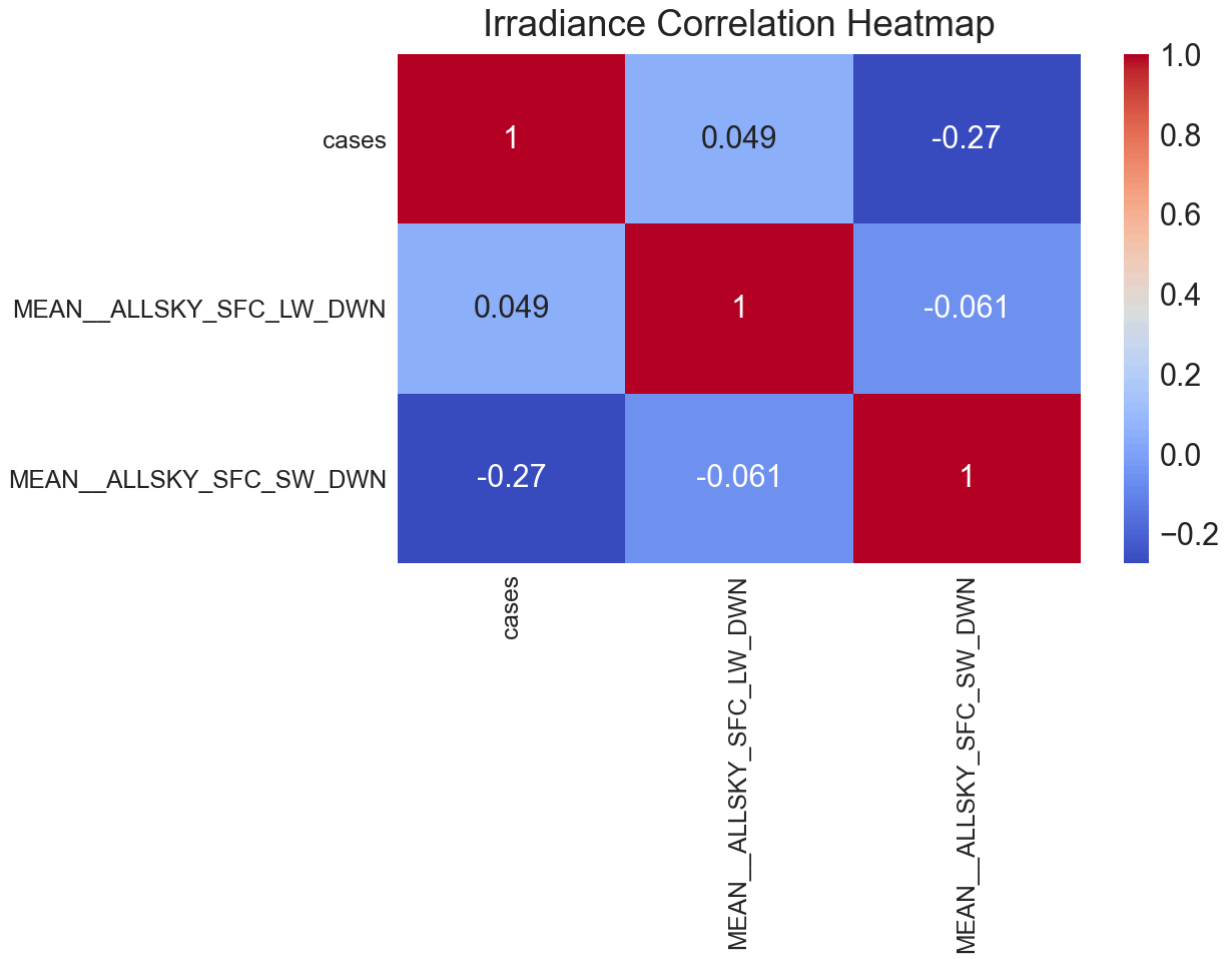
In [536]:
```
title = 'Soil Moisture'
corr_col = ['cases'] + soil_moisture
df_corr = df_all[corr_col].corr()

cols = len(corr_col)
plt.figure(figsize=(cols+2, cols))
ax = sns.heatmap(df_corr, annot=True, cmap='coolwarm')
ax.set_xticklabels(ax.get_xticklabels(), fontsize=8)  # Set x-axis label size
ax.set_yticklabels(ax.get_yticklabels(), fontsize=8)  # Set y-axis label size
plt.title(title + ' Correlation Heatmap')
plt.show()
```



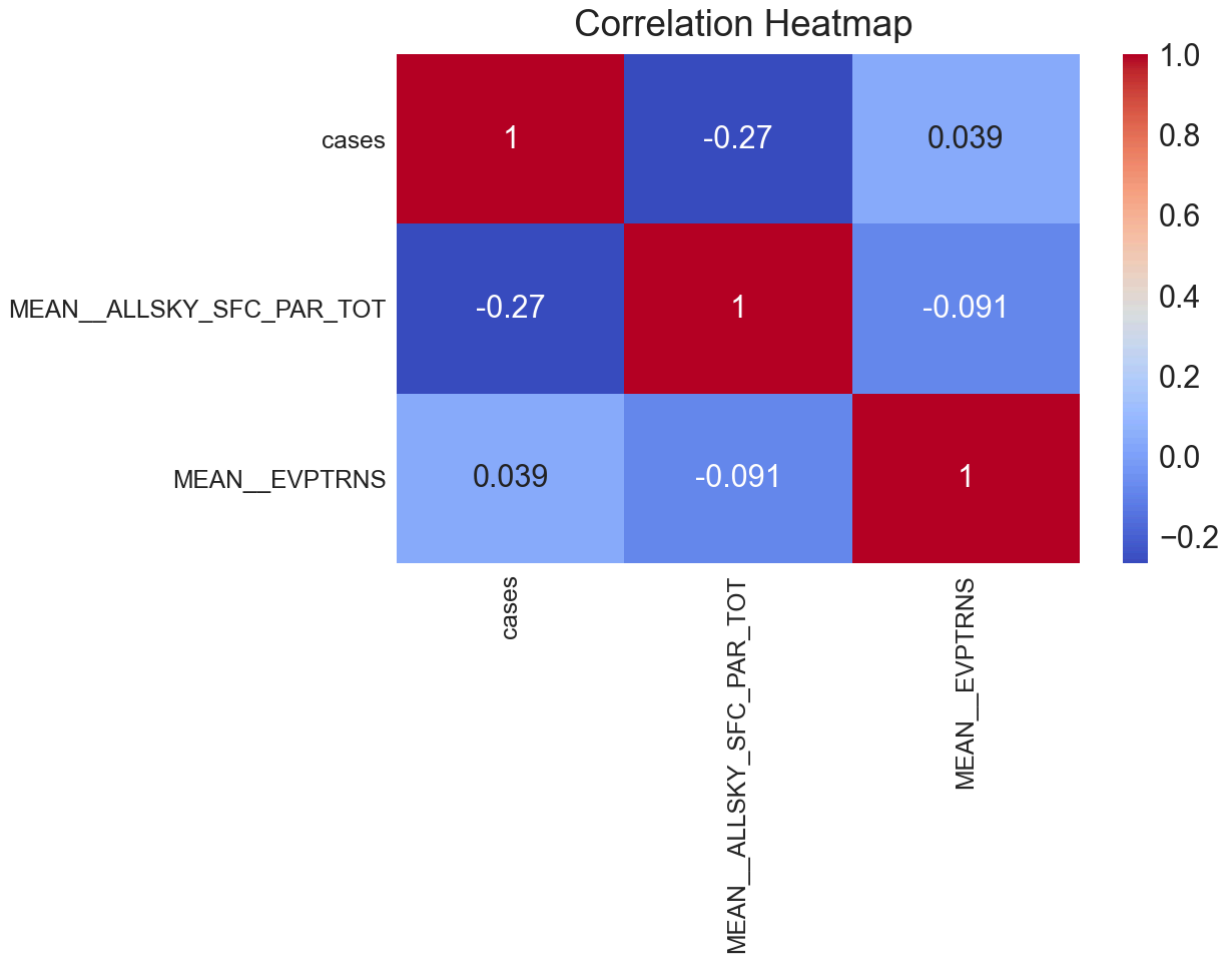
In [537]:
```
title = 'Temperature'
corr_col = ['cases'] + temperature
df_corr = df_all[corr_col].corr()

cols = len(corr_col)
plt.figure(figsize=(cols+2, cols))
ax = sns.heatmap(df_corr, annot=True, cmap='coolwarm')
ax.set_xticklabels(ax.get_xticklabels(), fontsize=8)  # Set x-axis label size
ax.set_yticklabels(ax.get_yticklabels(), fontsize=8)  # Set y-axis label size
plt.title(title + ' Correlation Heatmap')
plt.show()
```



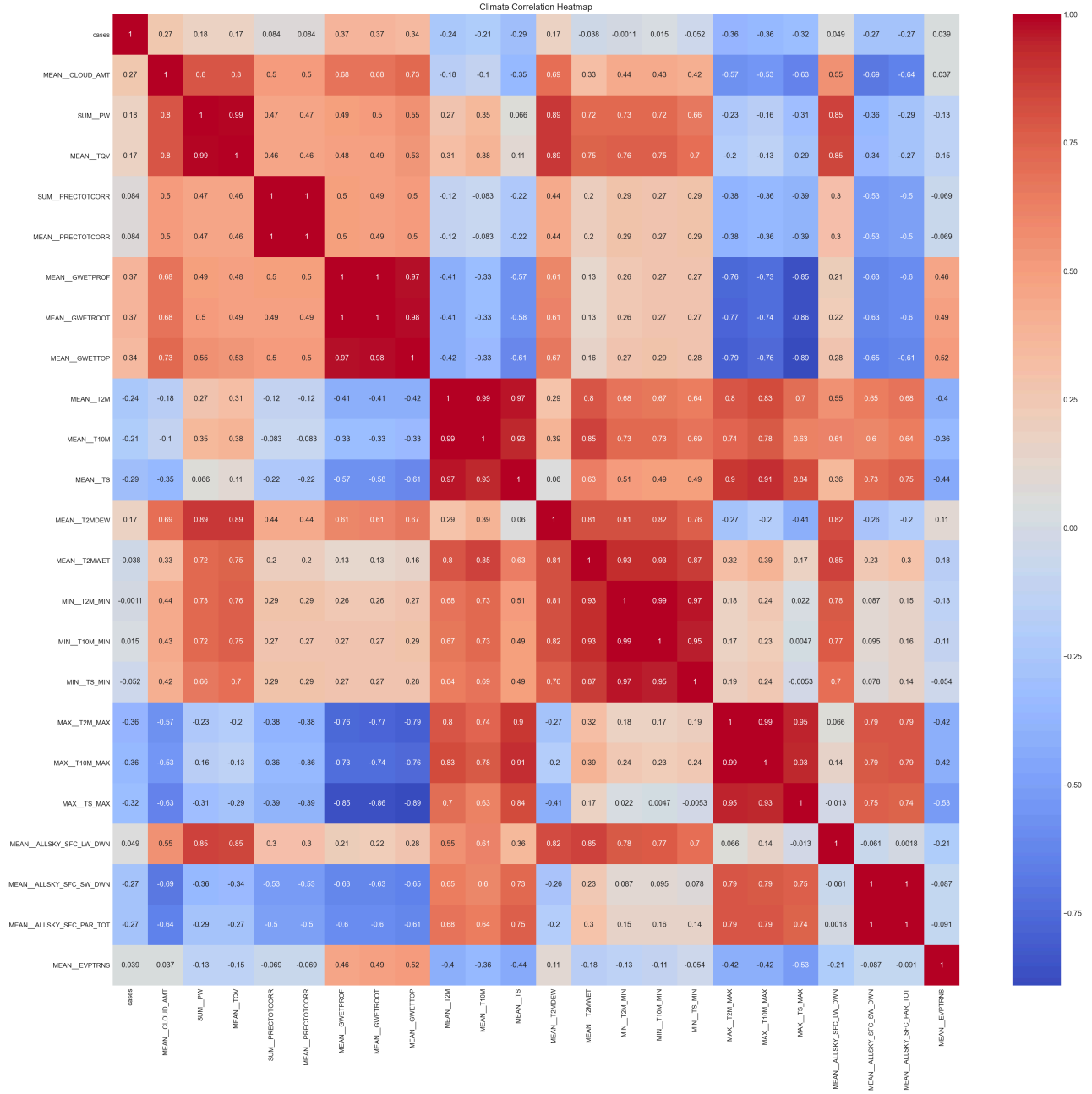
In [538]:
```
title = 'Irradiance'
corr_col = ['cases'] + irradiance
df_corr = df_all[corr_col].corr()

cols = len(corr_col)
plt.figure(figsize=(cols+2, cols))
ax = sns.heatmap(df_corr, annot=True, cmap='coolwarm')
ax.set_xticklabels(ax.get_xticklabels(), fontsize=8)  # Set x-axis label size
ax.set_yticklabels(ax.get_yticklabels(), fontsize=8)  # Set y-axis label size
plt.title(title + ' Correlation Heatmap')
plt.show()
```



In [539]:
```
title = ''
corr_col = ['cases'] + others
df_corr = df_all[corr_col].corr()

cols = len(corr_col)
plt.figure(figsize=(cols+2, cols))
ax = sns.heatmap(df_corr, annot=True, cmap='coolwarm')
ax.set_xticklabels(ax.get_xticklabels(), fontsize=8)  # Set x-axis label size
ax.set_yticklabels(ax.get_yticklabels(), fontsize=8)  # Set y-axis label size
plt.title(title + ' Correlation Heatmap')
plt.show()
```

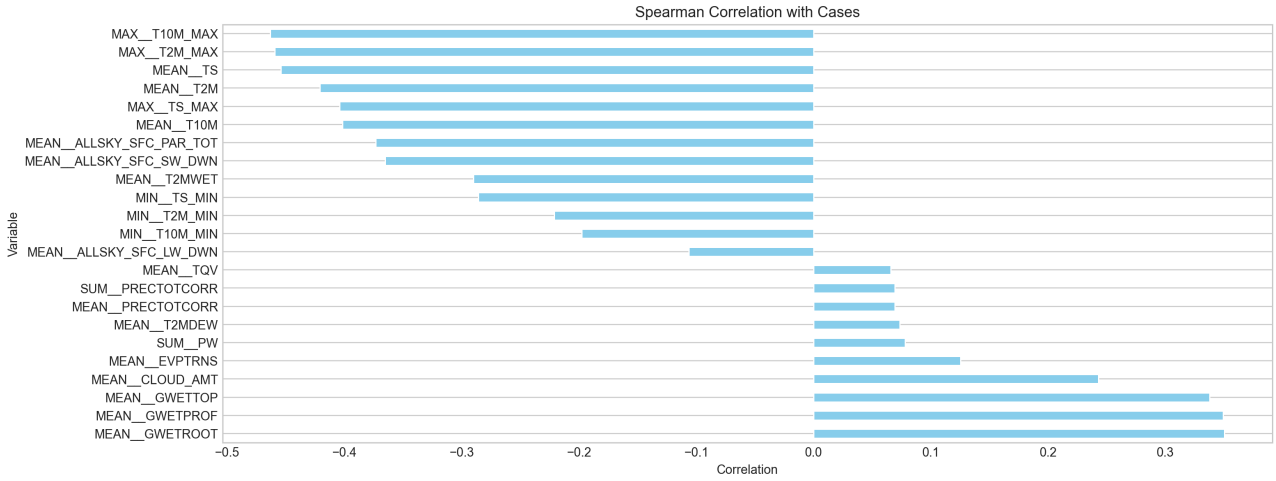


In [540]:
```
title = 'Climate'
corr_col = ['cases'] + climate
df_corr = df_all[corr_col].corr()

cols = len(corr_col)
plt.figure(figsize=(cols+2, cols))
ax = sns.heatmap(df_corr, annot=True, cmap='coolwarm')
ax.set_xticklabels(ax.get_xticklabels(), fontsize=9)  # Set x-axis label size
ax.set_yticklabels(ax.get_yticklabels(), fontsize=9)  # Set y-axis label size
plt.title(title + ' Correlation Heatmap')
plt.show()
```



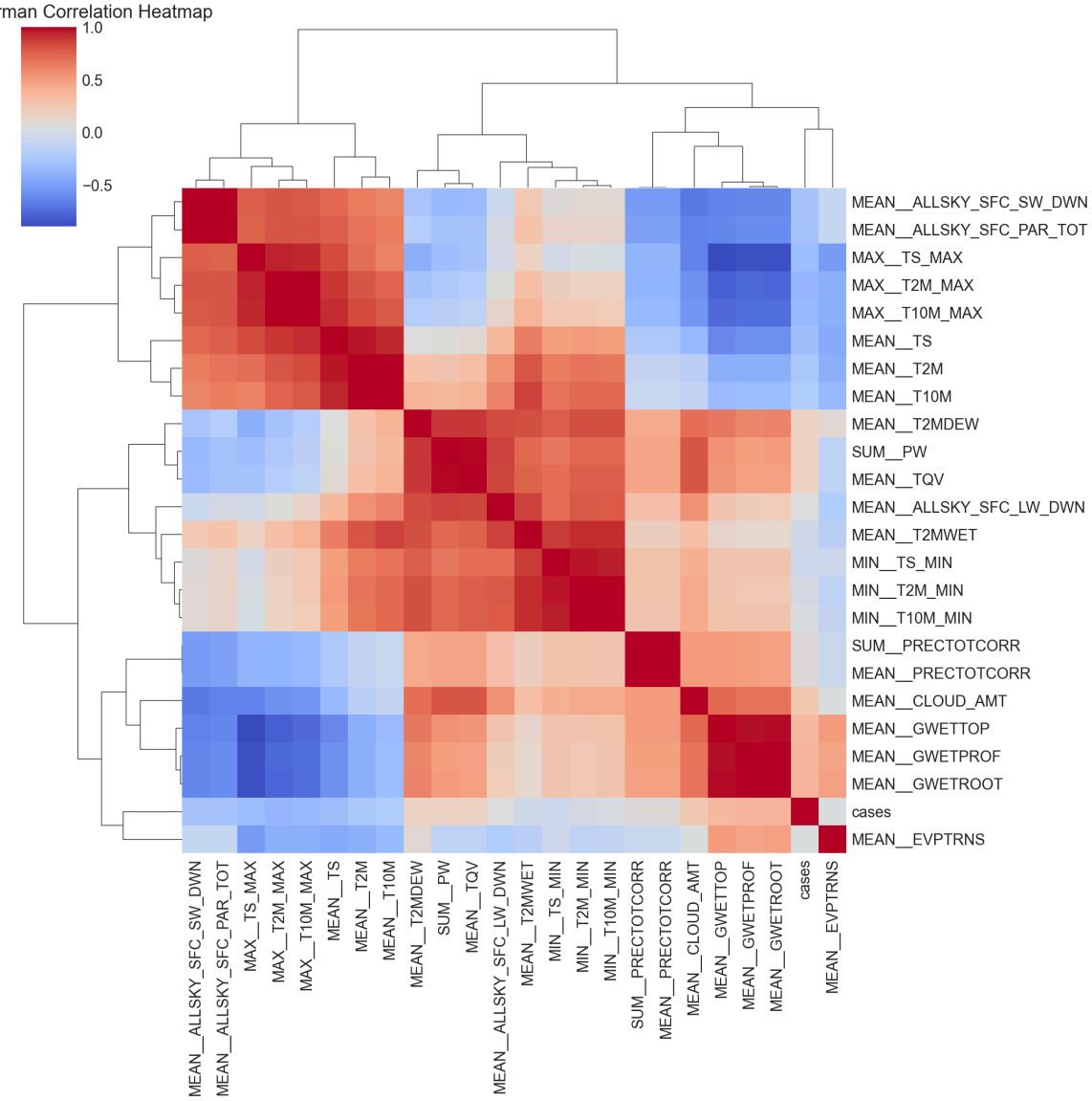
In [541]:
```
df_corr = df_all.drop(columns=['region', 'deaths', 'year', 'month'],axis=1).corr(method='spearman')
(df_corr.cases.drop('cases').sort_values(ascending=False).plot.barh(color='skyblue'))

plt.title('Spearman Correlation with Cases')
plt.xlabel('Correlation')
plt.ylabel('Variable')
plt.grid(axis='x')
plt.show()

title = 'Climate'
corr_col = ['cases'] + climate
df_corr = df_all[corr_col].corr()

cols = len(corr_col)
plt.figure(figsize=(cols+2, cols))
ax = sns.clustermap(df_corr,  cmap='coolwarm')
plt.title(title + ' Spearman Correlation Heatmap')
plt.show()
```

```
C:\Users\maryn\AppData\Local\Temp\ipykernel_27320\1302182115.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future versi
on, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  df_corr = df_all.drop(columns=['region', 'deaths', 'year', 'month'],axis=1).corr(method='spearman')
```



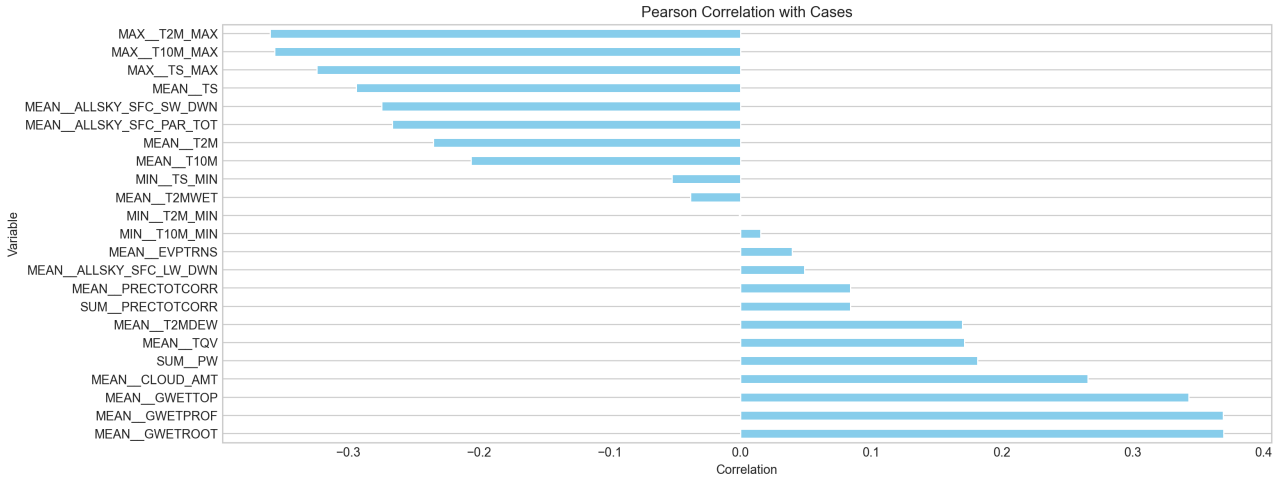

```
<Figure size 5720x5280 with 0 Axes>
```

Climate Spearman Correlation Heatmap



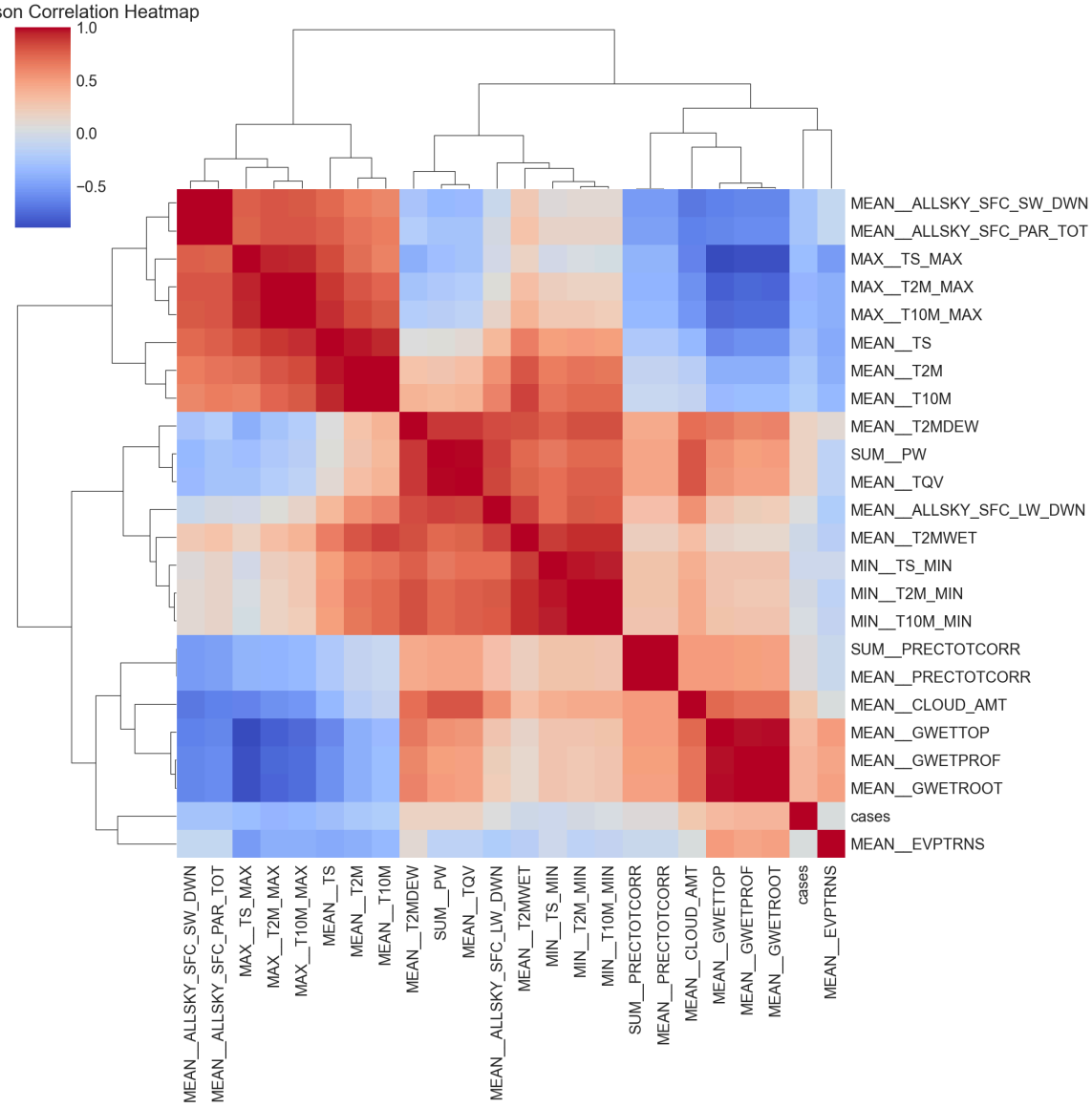
In [542]:
```
df_corr = df_all.drop(columns=['region', 'deaths', 'year', 'month'],axis=1).corr(method='pearson')
(df_corr.cases.drop('cases').sort_values(ascending=False).plot.barh(color='skyblue'))

plt.title('Pearson Correlation with Cases')
plt.xlabel('Correlation')
plt.ylabel('Variable')
plt.grid(axis='x')
plt.show()

title = 'Climate'
corr_col = ['cases'] + climate
df_corr = df_all[corr_col].corr()

cols = len(corr_col)
plt.figure(figsize=(cols+2, cols))
ax = sns.clustermap(df_corr,  cmap='coolwarm')
plt.title(title + ' Pearson Correlation Heatmap')
plt.show()
```

```
C:\Users\maryn\AppData\Local\Temp\ipykernel_27320\3480803446.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future versi
on, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  df_corr = df_all.drop(columns=['region', 'deaths', 'year', 'month'],axis=1).corr(method='pearson')
```



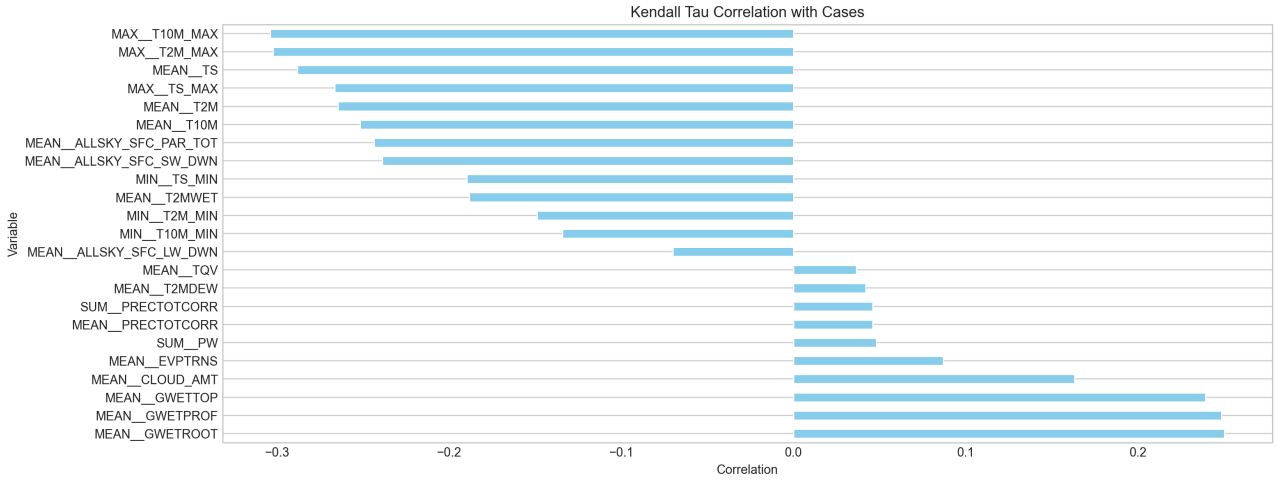

```
<Figure size 5720x5280 with 0 Axes>
```

Climate Pearson Correlation Heatmap

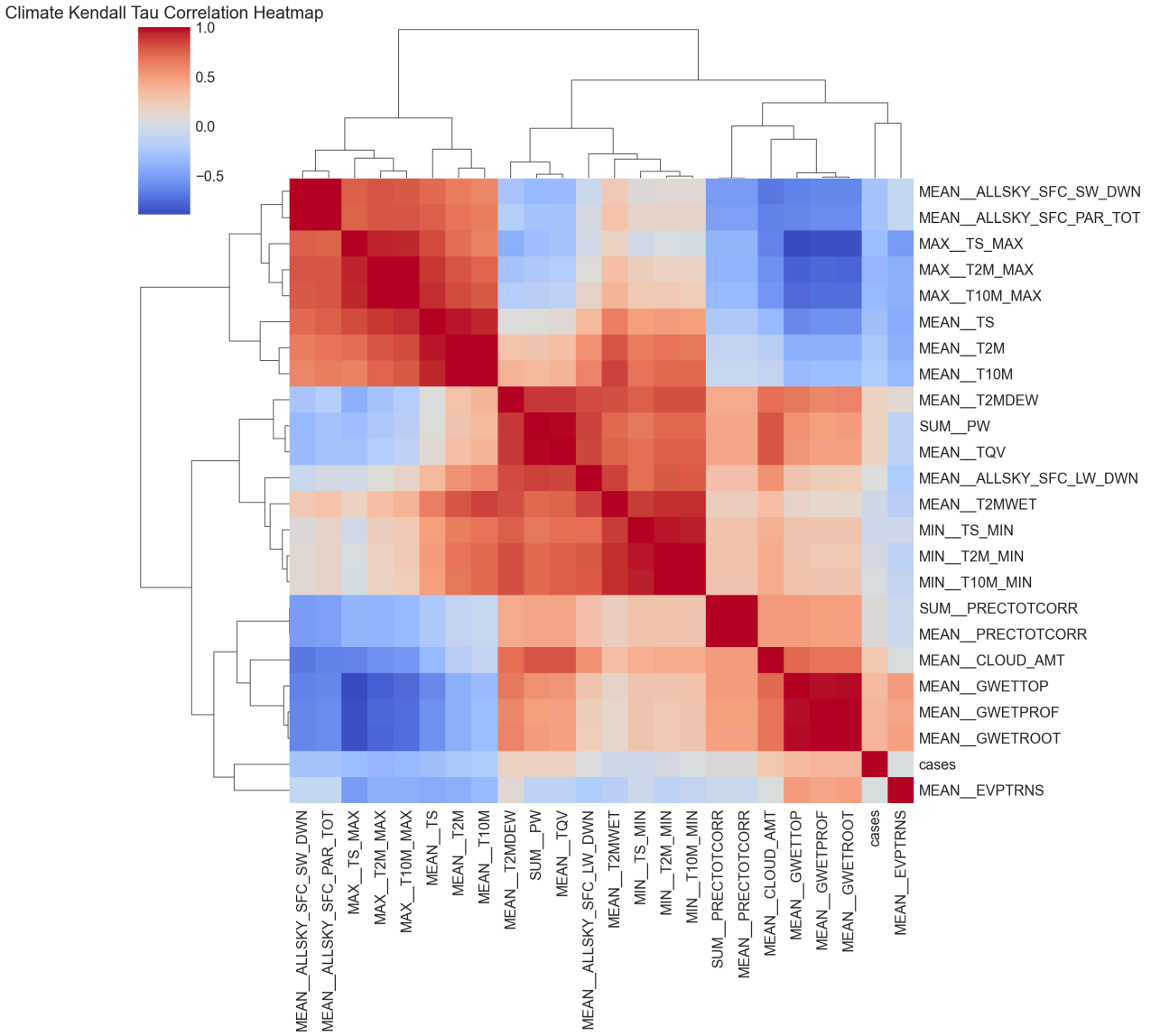


In [543]:
```
df_corr = df_all.drop(columns=['region', 'deaths', 'year', 'month'],axis=1).corr(method='kendall')
(df_corr.cases.drop('cases').sort_values(ascending=False).plot.barh(color='skyblue'))

plt.title('Kendall Tau Correlation with Cases')
plt.xlabel('Correlation')
plt.ylabel('Variable')
plt.grid(axis='x')
plt.show()

title = 'Climate'
corr_col = ['cases'] + climate
df_corr = df_all[corr_col].corr()

cols = len(corr_col)
plt.figure(figsize=(cols+2, cols))
ax = sns.clustermap(df_corr,  cmap='coolwarm')
plt.title(title + ' Kendall Tau Correlation Heatmap')
plt.show()
```

```
C:\Users\maryn\AppData\Local\Temp\ipykernel_27320\2931233373.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future versi
on, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  df_corr = df_all.drop(columns=['region', 'deaths', 'year', 'month'],axis=1).corr(method='kendall')
```




```
<Figure size 5720x5280 with 0 Axes>
```


Climate Kendall Tau Correlation Heatmap
1.0
0.5
0.0
−0.5
MEAN__ALLSKY_SFC_SW_DWN
MEAN__ALLSKY_SFC_PAR_TOT
MAX__TS_MAX
MAX__T2M_MAX
MAX__T10M_MAX
MEAN__TS
MEAN__T2M
MEAN__T10M
MEAN__T2MDEW
SUM__PW
MEAN__TQV
MEAN__ALLSKY_SFC_LW_DWN
MEAN__T2MWET
MIN__TS_MIN
MIN__T2M_MIN
MIN__T10M_MIN
SUM__PRECTOTCORR
MEAN__PRECTOTCORR
MEAN__CLOUD_AMT
MEAN__GWETTOP
MEAN__GWETPROF
MEAN__GWETROOT
cases
MEAN__EVPTRNS

In [ ]: