Google Colab Link: [https://colab.research.google.com/drive/12PS0_uLqe11SQY7Q0L2WBnMVfYjrHSpc?usp=sharing (https://colab.research.google.com/drive/12PS0_uLqe11SQY7Q0L2WBnMVfYjrHSpc?usp=sharing)](https://colab.research.google.com/drive/12PS0_uLqe11SQY7Q0L2WBnMVfYjrHSpc?usp=sharing)

In [1]:
```
# Importing all necessary packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import norm,stats,chi2_contingency,levene,kruskal,shapiro,
f_oneway,ttest_ind
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
# Importing the data
raw_df = pd.read_csv("/content/drive/MyDrive/Dataset/bike_sharing.csv")
```

In [3]:
```
#checking the data
raw_df.head()
```

Out[3]:

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casu
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.0	
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0	
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0	
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0	
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0	

In [4]:
```
# Shape of the data
raw_df.shape
```

Out[4]:
```
(10886, 12)
```

Column Details

- **datetime:** The date and time of the observation.
- **season:** The season of the year, represented numerically.

1. Spring
2. Summer
3. Fall
4. Winter

- **holiday**: Indicates whether the day is a holiday (1) or not (0).
- **workingday**: Indicates whether the day is a working day (1) or not (0). A working day is defined as neither a weekend nor a holiday.
- **weather:** Describes the weather conditions.

1. Clear, Few clouds, Partly cloudy
2. Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds
3. Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds
4. Heavy Rain + Ice Pellets + Thunderstorm + Mist, Snow + Fog

- **temp:** The temperature in Celsius.
- **atemp:** The "feels like" temperature in Celsius.
- **humidity:** The humidity level.
- **windspeed:** The wind speed.
- **casual:** The count of casual users.
- **registered:** The count of registered users.
- **count:** The total count of rental bikes, including both casual and registered users.

In [5]:

```
# Null Data check
raw_df.isna().sum()
```

Out[5]:

```
datetime      0
season        0
holiday       0
workingday    0
weather       0
temp          0
atemp         0
humidity      0
windspeed     0
casual        0
registered    0
count         0
dtype: int64
```

In [6]:

```
# Creating few columns from datetime field for better analysis of the data.
raw_df['datetime']=pd.to_datetime(raw_df['datetime'])

raw_df["year"] = raw_df["datetime"].dt.year
raw_df["month"] = raw_df["datetime"].dt.month
raw_df["day"] = raw_df["datetime"].dt.day
raw_df["hour_of_the_day"] = raw_df["datetime"].dt.hour
raw_df['quarter'] = raw_df['datetime'].dt.quarter

# Dropping the datetime column, as the required data has been extracted int
o different columns, and the columns serves no purpose
raw_df.drop(["datetime"],axis=1, inplace=True)

# Modifying the data type of following few features, as per the understandi
ng, they are categorical data.
raw_df["season"] = raw_df["season"].astype("object")
raw_df["holiday"] = raw_df["holiday"].astype("object")
raw_df["workingday"] = raw_df["workingday"].astype("object")
raw_df["weather"] = raw_df["weather"].astype("object")
raw_df["year"] = raw_df["year"].astype("object")
raw_df["month"] = raw_df["month"].astype("object")
raw_df["day"] = raw_df["day"].astype("object")
raw_df["hour_of_the_day"] = raw_df["hour_of_the_day"].astype("object")
raw_df["quarter"] = raw_df["quarter"].astype("object")
```

In [7]:

```
# Description of Data (Data Type)
raw_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10886 entries, 0 to 10885
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   season           10886 non-null  object 
 1   holiday          10886 non-null  object 
 2   workingday       10886 non-null  object 
 3   weather          10886 non-null  object 
 4   temp             10886 non-null  float64
 5   atemp            10886 non-null  float64
 6   humidity         10886 non-null  int64  
 7   windspeed        10886 non-null  float64
 8   casual           10886 non-null  int64  
 9   registered       10886 non-null  int64  
 10  count            10886 non-null  int64  
 11  year             10886 non-null  object 
 12  month            10886 non-null  object 
 13  day              10886 non-null  object 
 14  hour_of_the_day  10886 non-null  object 
 15  quarter          10886 non-null  object 
dtypes: float64(3), int64(4), object(9)
memory usage: 1.3+ MB
```

In [8]:

```python
# Description of Data (Categorical Data Description)
raw_df.describe(include= object).T
```

Out[8]:

	count	unique	top	freq
season	10886	4	4	2734
holiday	10886	2	0	10575
workingday	10886	2	1	7412
weather	10886	4	1	7192
year	10886	2	2012	5464
month	10886	12	5	912
day	10886	19	1	575
hour_of_the_day	10886	24	12	456
quarter	10886	4	4	2734

In [9]:

```python
# Description of Data (Continuous Data Description)
raw_df.describe().T
```

Out[9]:

	count	mean	std	min	25%	50%	75%	max
temp	10886.0	20.230860	7.791590	0.82	13.9400	20.500	26.2400	41.0000
atemp	10886.0	23.655084	8.474601	0.76	16.6650	24.240	31.0600	45.4550
humidity	10886.0	61.886460	19.245033	0.00	47.0000	62.000	77.0000	100.0000
windspeed	10886.0	12.799395	8.164537	0.00	7.0015	12.998	16.9979	56.9969
casual	10886.0	36.021955	49.960477	0.00	4.0000	17.000	49.0000	367.0000
registered	10886.0	155.552177	151.039033	0.00	36.0000	118.000	222.0000	886.0000
count	10886.0	191.574132	181.144454	1.00	42.0000	145.000	284.0000	977.0000

Analysis and Explanation

We analyzed the Yulu rented bikes dataset comprising 10,886 data points across 12 features, with no missing values. Additionally, we derived 5 features from the datetime column. Here are the key observations:

1. **Season and Weather Distribution:**

- The dataset covers 4 seasons and 4 types of weather. The majority of data points are in season 4 (winter) and weather type 1 (clear or cloudy). This suggests a consumer preference for using rented bikes during winter months under clear or cloudy weather conditions, possibly to enjoy the winter sun.

1. **Temporal Distribution:**

- The data spans from 2011 to 2012, with a higher concentration in 2012 (5,464 records). Most data points are in the final quarter of the year, aligning with the preference for winter rentals. Peak rental times are around noon, further emphasizing the preference for clear skies during the day.

1. **Monthly and Daily Trends:**

- Although winter sees the most rentals overall, May stands out as the busiest month. Interestingly, the 1st of each month records significant rental activity, indicating a potential trend or promotional activity on these days.

1. **Working Day vs. Holiday Rentals:**

- The majority of rentals occur on working days rather than holidays, suggesting that users predominantly utilize the bikes for commuting to workplaces.

1. **Temperature and Weather Conditions:**

- Temperature ranges from 0.82°C to 41°C, with an average of approximately 21°C. The "feels like" temperature (atemp) ranges from 0.76°C to 45.45°C, with an average of 23.65°C. Humidity spans from 0 to 100, with an average of 61.9. Windspeed ranges from 0 to 57 (unit unspecified), with an average of 12.8.

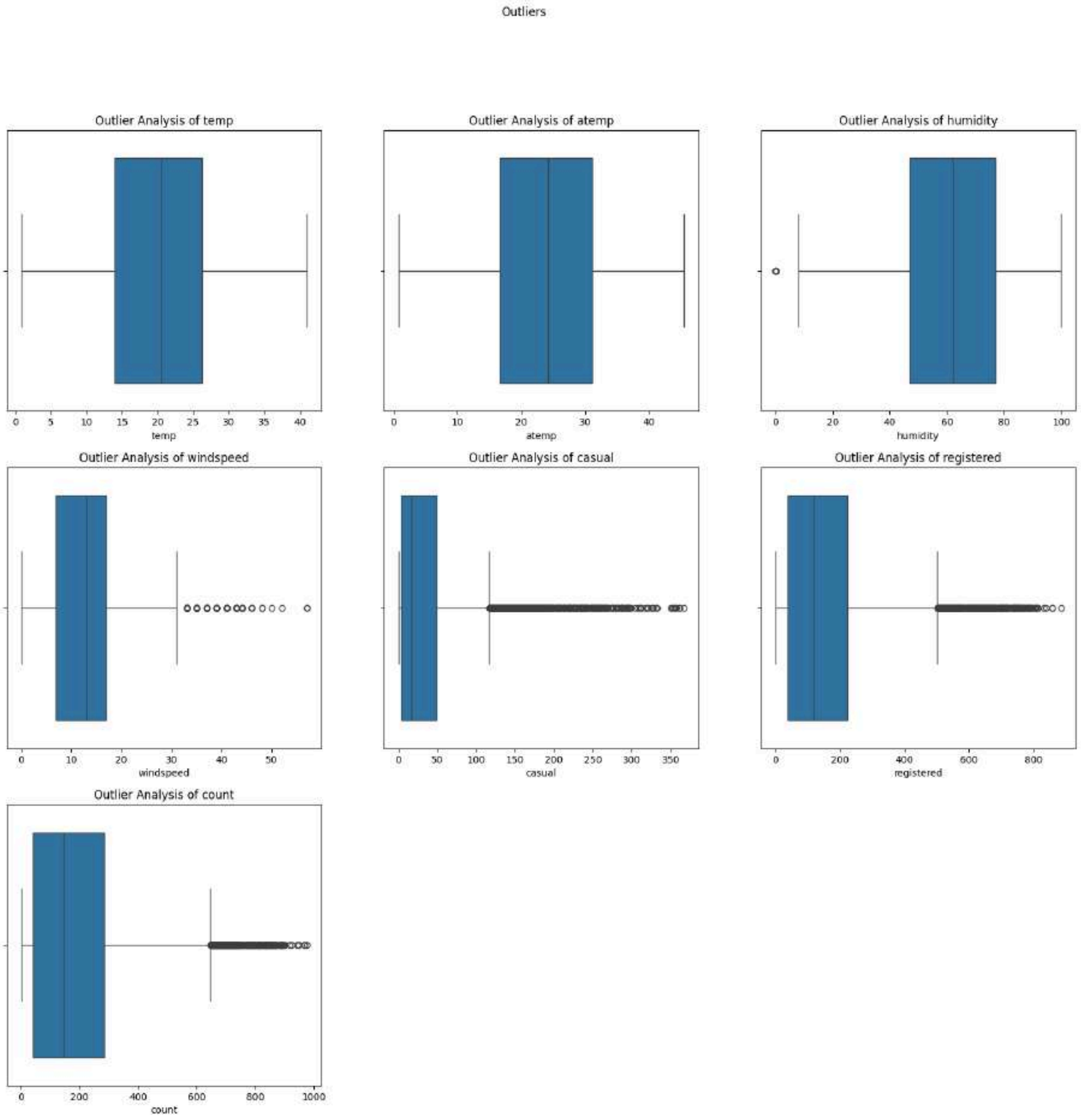
1. **Bike Rental Counts:**

- The count of rented bikes per day ranges from 0 to 977, with an average of approximately 192 rentals daily. Casual ridership is lower, ranging from 0 to 367 with an average of 36 riders, while registered users show a wider range (0 to 886) with an average of 151 riders per day.

Outlier Detection and Treatment

In [10]:

```
# Plotting boxplot to detect the outliers in all quantative data and check if we need to remove them.

plt.figure(figsize = (20,18))
plt.suptitle("Outliers")
features = ['temp', 'atemp','humidity', 'windspeed', 'casual', 'registered', 'count']
for i in range(len(features)):
     plt.subplot(3, 3, i+1)
     sns.boxplot(x = raw_df[features[i]])
     plt.title('Outlier Analysis of {}'.format(features[i]))

plt.show()
```



It seems outliers are present on windspeed,casual,registered,count.

In [11]:

```
# Creating a function to provide the details of whiskers and quartile range for all columns provided of the dataframe

def outlier_info(df,columns_list):
    for col in columns_list:
        print("\nOutlier data for {}".format(col))
        Q1 = df[col].quantile(0.25)
        Q2 = df[col].median()
        Q3 = df[col].quantile(0.75)
        IQR = Q3 - Q1
        lower_whisker = Q1 - (1.5 * IQR)
        upper_whisker = Q3 + (1.5 * IQR)
        print("Lower Whisker(Income): {} \nQuartile-1 : {}\nQuartile-2 : {}\nQuartile-3 : {}\nIQR : {}\nUpper Whisker(Income) : {}".format(lower_whisker,Q1,Q2,Q3,IQR,upper_whisker))
```

In [12]:

```
# Invoking the above function and provide the dataframe and list of columns we just detected with outliers

outlier_info(raw_df,["windspeed","casual","registered","count"])
```

```
Outlier data for windspeed
Lower Whisker(Income): -7.993100000000002
Quartile-1 : 7.0015
Quartile-2 : 12.998
Quartile-3 : 16.9979
IQR : 9.996400000000001
Upper Whisker(Income) : 31.992500000000003

Outlier data for casual
Lower Whisker(Income): -63.5
Quartile-1 : 4.0
Quartile-2 : 17.0
Quartile-3 : 49.0
IQR : 45.0
Upper Whisker(Income) : 116.5

Outlier data for registered
Lower Whisker(Income): -243.0
Quartile-1 : 36.0
Quartile-2 : 118.0
Quartile-3 : 222.0
IQR : 186.0
Upper Whisker(Income) : 501.0

Outlier data for count
Lower Whisker(Income): -321.0
Quartile-1 : 42.0
Quartile-2 : 145.0
Quartile-3 : 284.0
IQR : 242.0
Upper Whisker(Income) : 647.0
```

Interpretations of the Above Outcomes

Outliers Detection Criteria:

- Data points where windspeed exceeds 31.99 are considered outliers. This threshold is used to identify extreme wind conditions that may influence bike rental patterns.
- Casual rider counts exceeding 116.5 are flagged as outliers. This threshold helps identify unusually high casual rider activity days.
- Similarly, registered rider counts greater than 501 are identified as outliers. This criterion identifies days with exceptionally high registered rider demand.
- The total count of riders exceeding 647 on a given day is marked as an outlier. This combines both casual and registered riders to identify days with unusually high overall bike rental activity.

These outlier thresholds provide a framework to understand and potentially mitigate extreme data points that could skew analyses or models. Outlier identification helps in refining statistical analyses and ensuring more accurate insights into bike rental patterns.

In [13]:

```python
# Taking the count of records that are considered as outliers as per the above conditions

print("Count of datapoints on column 'Windspeed' is {}.".format(raw_df.loc[raw_df["windspeed"] > 31.99,["season"]].count()["season"]))
print("Count of datapoints on column 'casual' is {}.".format(raw_df.loc[raw_df["casual"] > 31.99,["season"]].count()["season"]))
print("Count of datapoints on column 'registered' is {}.".format(raw_df.loc[raw_df["registered"] > 31.99,["season"]].count()["season"]))
print("Count of datapoints on column 'count' is {}.".format(raw_df.loc[raw_df["count"] > 31.99,["season"]].count()["season"]))
```

```
Count of datapoints on column 'Windspeed' is 227.
Count of datapoints on column 'casual' is 3818.
Count of datapoints on column 'registered' is 8323.
Count of datapoints on column 'count' is 8536.
```

In [14]:

```
# creating a function to assist analyzing the data if outliers are removed vs if they were not removed, by passing the dataframe and a dictionary with column name and their threshold values

def before_after_outlier_analysis(df,columns_dict):
    for columns,threshold in columns_dict.items():
        print("\nOutlier Impact Analysis on data for {}".format(columns))

       # Before
        before_mean = df[columns].mean()
        before_stddev = df[columns].std()
        before_median = df[columns].median()

        print("If outliers are NOT removed \n \nMean: {} \nStandard Deviation : {}\nMedian : {}\n".format(before_mean,before_stddev,before_median))

        # If outliers are removed

        after_mean = df.loc[df[columns] < threshold][columns].mean()
        after_stddev = df.loc[df[columns] < threshold][columns].std()
        after_median = df.loc[df[columns] < threshold][columns].median()

        print("If outliers are removed \n \nMean: {} \nStandard Deviation : {}\nMedian : {}".format(after_mean,after_stddev,after_median))
        print("---------------------------------------------------------------------------------------------------------")
```

In [15]:

```
# Creating a dictionary to pass to the above written functions as parameters, and invoking the same.

columns_dict = {
    "windspeed":31.99,
    "casual":116.5,
    "registered":501,
    "count":647
}

before_after_outlier_analysis(raw_df,columns_dict)
```

```
Outlier Impact Analysis on data for windspeed
If outliers are NOT removed

Mean: 12.7993954069447
Standard Deviation : 8.164537326838689
Median : 12.998

If outliers are removed

Mean: 12.292751927948213
Standard Deviation : 7.441015147553967
Median : 11.0014
---------------------------------------------------------------------------
------------------------------

Outlier Impact Analysis on data for casual
If outliers are NOT removed

Mean: 36.02195480433584
Standard Deviation : 49.960476572649526
Median : 17.0

If outliers are removed

Mean: 25.241984808128638
Standard Deviation : 27.93705825036805
Median : 14.0
---------------------------------------------------------------------------
------------------------------

Outlier Impact Analysis on data for registered
If outliers are NOT removed

Mean: 155.5521771082124
Standard Deviation : 151.03903308192454
Median : 118.0

If outliers are removed

Mean: 136.28264194226725
Standard Deviation : 117.84035725136775
Median : 112.0
---------------------------------------------------------------------------
------------------------------

Outlier Impact Analysis on data for count
If outliers are NOT removed

Mean: 191.57413191254824
Standard Deviation : 181.14445383028527
Median : 145.0

If outliers are removed

Mean: 175.5834829443447
Standard Deviation : 156.1806722380325
Median : 138.0
---------------------------------------------------------------------------
------------------------------
```

Interpretations on the Above Outcome

1. **Impact of Outliers on Windspeed:**

- Removing data points where windspeed exceeds 31.99 does not significantly alter the average windspeed in the dataset. Therefore, it is concluded that retaining these outliers does not distort the overall analysis related to windspeed.

1. **Impact of Outliers on Casual, Registered, and Total Counts:**

- Outliers in the columns for casual riders, registered riders, and total rental counts (casual + registered) have a noticeable effect on their respective averages and standard deviations. This indicates that outlier values disproportionately influence these metrics.

1. **Actionable Insight:**

- Given the substantial impact of outliers on casual, registered, and total counts, it is recommended to remove data points that exceed the identified outlier thresholds in these columns. By doing so, the analysis can focus on the more typical patterns of bike rental behavior without the distortion caused by extreme values.

In [16]:

```
# Removing the outliers data and copying the data into another dataframe, t
hat can be considered as final_df

final_df = raw_df[ ~(raw_df['casual']> 116.5) ]
final_df = final_df[ ~(final_df['registered']> 501) ]
final_df = final_df[ ~(final_df['count']> 647) ]

final_df.reset_index(drop=True, inplace=True)
```

Correlation Analysis

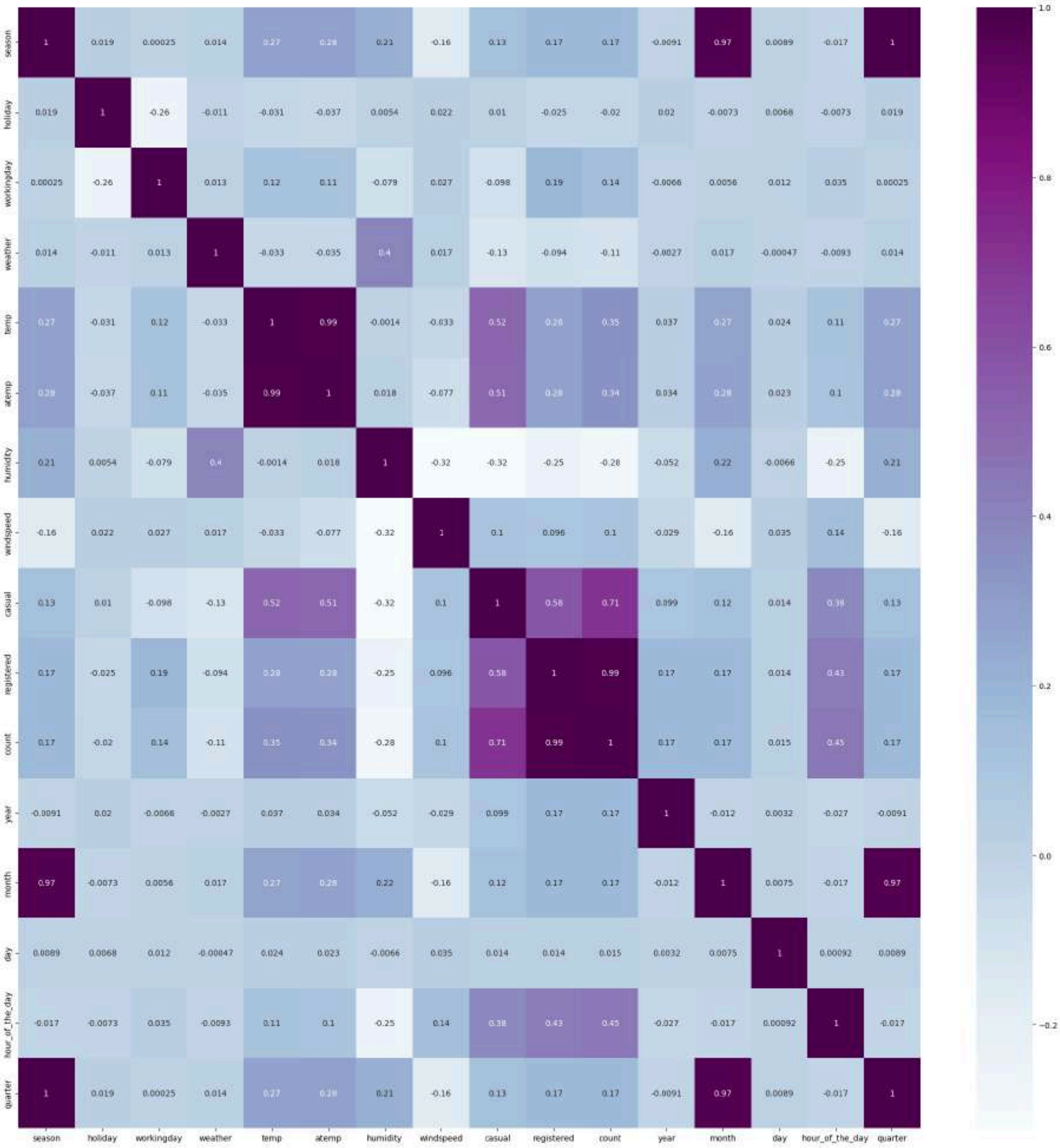
In [17]:

```
# Converting the categorical data to numerical for correlation analysis

final_df["season"] = final_df["season"].astype("int")
final_df["holiday"] = final_df["holiday"].astype("int")
final_df["workingday"] = final_df["workingday"].astype("int")
final_df["weather"] = final_df["weather"].astype("int")
final_df["year"] = final_df["year"].astype("int")
final_df["month"] = final_df["month"].astype("int")
final_df["day"] = final_df["day"].astype("int")
final_df["hour_of_the_day"] = final_df["hour_of_the_day"].astype("int")
final_df["quarter"] = final_df["quarter"].astype("int")
```

Pearson Method

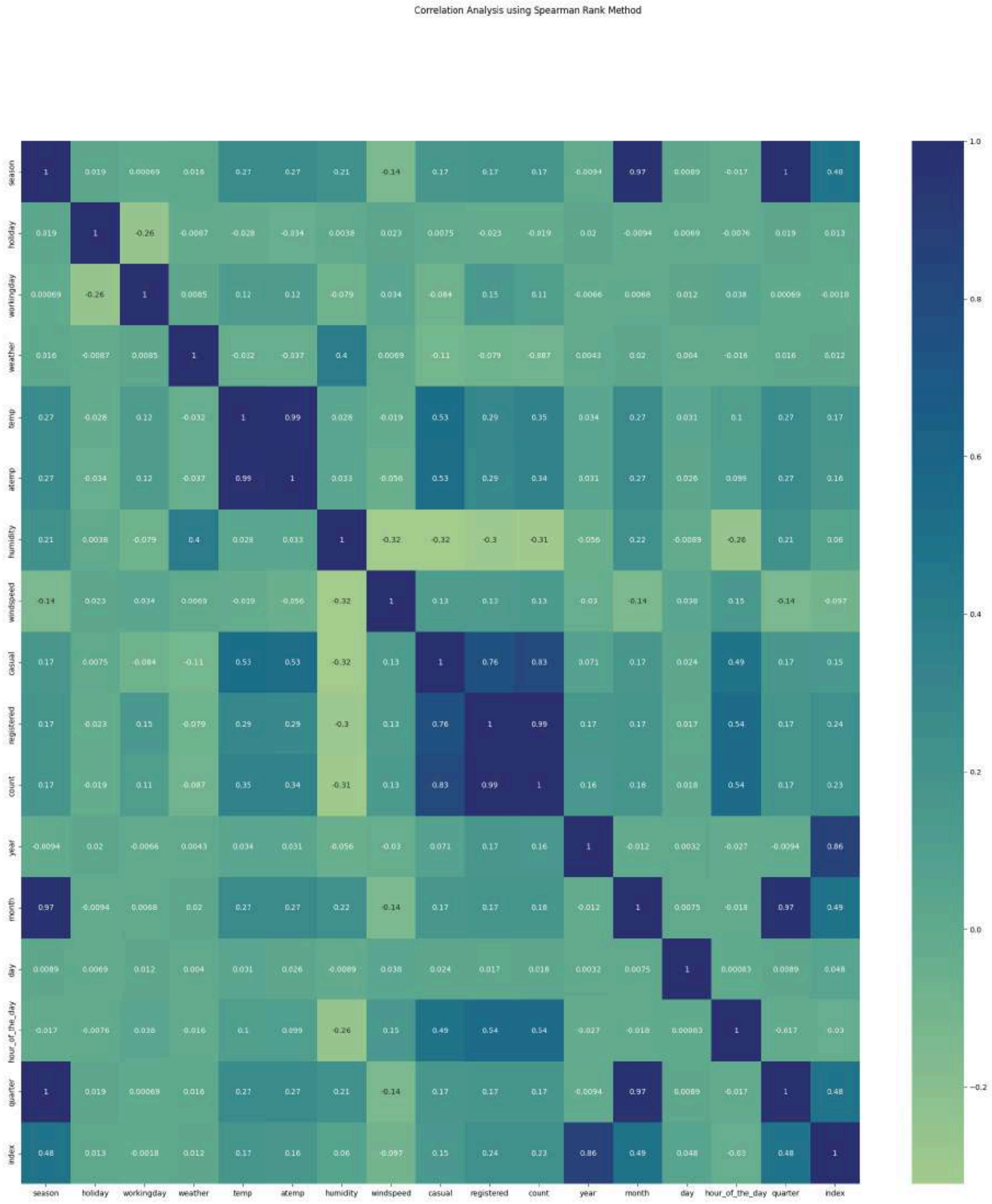
In [18]:
```
# Plotting heatmap on the correlation analysis done using Pearson method

fig = plt.figure(figsize=(25,25))
fig.suptitle("Correlation Analysis using Pearson Method")
sns.heatmap(final_df.corr(),annot=True,cmap="BuPu")
plt.show()
```

Correlation Analysis using Pearson Method



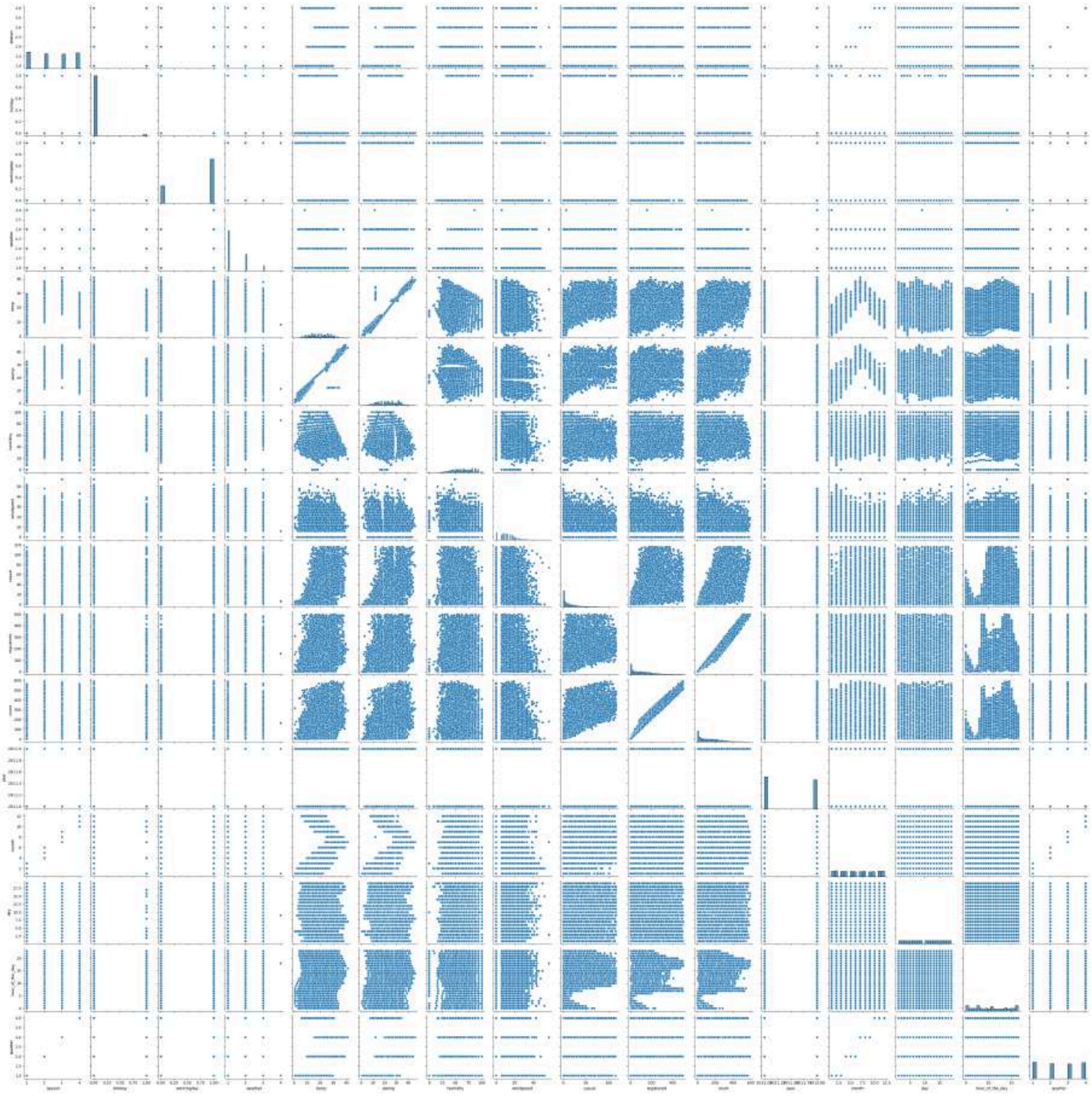
Spearman Rank Method

In [66]:
```
# Plotting heatmap on the correlation analysis done using Spearman method

fig = plt.figure(figsize=(25,25))
fig.suptitle("Correlation Analysis using Spearman Rank Method")
sns.heatmap(final_df.corr(method="spearman"),annot=True,cmap="crest")
plt.show()
```



In [20]:
```
plt.figure(figsize=(20,10))
sns.pairplot(final_df)
plt.show()
```

```
<Figure size 2000x1000 with 0 Axes>
```



Analysis and Explanation

We utilized both Pearson and Spearman correlation methods to assess relationships among the features, complemented by heatmaps and pair plots. Overall, the dataset exhibits minimal to negligible correlations between most variables.

Key Observations:

1. **Strong Positive Correlations:**

- **Season with Month and Quarter:** There is a strong positive correlation between the season and both the month and quarter. This relationship is expected, as each season spans several months.
- **Temperature with Felt Temperature:** Temperature and felt temperature show a strong positive correlation, which is intuitive as felt temperature is derived from temperature and factors like humidity and wind speed.

1. **Weak Positive Correlations:**

- **Casual Riders with Temperature and Felt Temperature:** There is a weak positive correlation between the count of casual riders and both temperature and felt temperature. This suggests that warmer temperatures may slightly influence casual ridership.

1. **General Correlation Insights:**

- The analysis reveals that most features in the dataset are not strongly correlated. This indicates that bike rental patterns are likely influenced by a combination of factors rather than a single dominant variable.

In [22]:

```
# Converting the columns that were converted earlier, back to categories

final_df["season"] = final_df["season"].astype("object")
final_df["holiday"] = final_df["holiday"].astype("object")
final_df["workingday"] = final_df["workingday"].astype("object")
final_df["weather"] = final_df["weather"].astype("object")
final_df["year"] = final_df["year"].astype("object")
final_df["month"] = final_df["month"].astype("object")
final_df["day"] = final_df["day"].astype("object")
final_df["hour_of_the_day"] = final_df["hour_of_the_day"].astype("object")
final_df["quarter"] = final_df["quarter"].astype("object")
```

Non-Graphical Analysis - Value Counts and Unique Attributes

In [23]:

```
#  Creating an identity column as primary key that will act as unique identifier for better analysis
final_df["index"] = range(1, final_df.shape[0] + 1)
```

In [25]:
```
# Count of data as per the season

count_matrix = final_df.groupby(["season"])["index"].count().sort_values(ascending=False).reset_index()
count_matrix.columns=["season","count_of_data_points"]
count_matrix
```

Out[25]:

	season	count_of_data_points
0	1	2600
1	4	2493
2	2	2346
3	3	2305

In [24]:
```
# Statistical Central Tendency analysis of data as per the season

count_matrix = final_df.groupby(["season"])[["temp","atemp","humidity","windspeed","casual","registered","count"]].mean().reset_index()
count_matrix.columns=["season","Average Temperature","Average Feeling Temperature","Average Humidity","Average Windspeed","Average Casual Bike Rents","Average Regisetered Bike Rents","Average Count of Bike Rents"]
count_matrix
```

Out[24]:

	season	Average Temperature	Average Feeling Temperature	Average Humidity	Average Windspeed	Average Casual Bike Rents	Average Regisetered Bike Rents	Aver Coun Bike Re
0	1	12.270354	14.929667	56.466538	14.596883	11.706154	91.553077	103.259
1	2	22.330844	26.134740	63.380222	13.245712	30.348679	130.614237	160.962
2	3	28.394768	32.134594	66.183514	11.135633	35.270716	142.140998	177.411
3	4	16.299021	19.672012	67.446049	11.488538	20.165263	142.418371	162.583

Analysis and Explanation

Following the removal of outliers, we observed the distribution of records across different seasons and analyzed the average counts of casual riders, registered riders, and total riders (casual + registered) across seasons.

Seasonal Distribution Post Outlier Removal:

1. **Uniform Distribution Across Seasons:**

- After removing outliers, each season shows a similar count of records, with slightly higher data points observed in the spring season compared to other seasons. This suggests a balanced dataset across different seasons post-cleaning.

Average Counts Across Seasons:

1. **Casual Riders and Total Riders:**

- **Fall Season:** The average count of casual riders and total riders is higher during the fall season. This indicates that casual users tend to utilize bikes more frequently during autumn months.

1. **Registered Riders:**

- **Winter Season:** On average, registered riders rented slightly more bikes during the winter season compared to fall. This suggests that registered users show a preference for winter rentals over fall, potentially due to weather conditions or commuting patterns.

In [26]:

```
# Count of data as per the Holidays

count_matrix = final_df.groupby(["holiday"])["index"].count().sort_values(ascending=False).reset_index()
count_matrix.columns=["Holiday","count_of_data_points"]
count_matrix
```

Out[26]:

	Holiday	count_of_data_points
0	0	9484
1	1	260

In [27]:

```
# Statistical Central Tendency analysis of data as per the holidays

count_matrix = final_df.groupby(["holiday"])[["temp","atemp","humidity","wi
ndspeed","casual","registered","count"]].mean().reset_index()
count_matrix.columns=["Holiday","Average Temperature","Average Feeling Temp
erature","Average Humidity","Average Windspeed","Average Casual Bike Rent
s","Average Regisetered Bike Rents","Average Count of Bike Rents"]
count_matrix
```

Out[27]:

	Holiday	Average Temperature	Average Feeling Temperature	Average Humidity	Average Windspeed	Average Casual Bike Rents	Average Regisetered Bike Rents	Aver Cour Bike Re
0	0	19.577630	22.961647	63.221636	12.627670	23.888444	126.409321	150.297
1	1	18.077846	21.052077	63.865385	13.746442	25.565385	108.757692	134.323

Analysis and Explanation

Following the removal of outliers from the dataset, we examined the distribution of data points between non-holiday and holiday periods, and analyzed the average counts of casual riders, registered riders, and total riders (casual + registered) for each category.

Distribution Between Non-Holiday and Holiday Periods Post Outlier Removal:

1. **Consistent Trend in Data Distribution:**

- Similar to the initial observation, the dataset continues to show a higher count of data points on non-holiday days compared to holidays. This reaffirms the trend that more bike rentals occur on regular weekdays and weekends, which are non-holiday periods.

Average Counts Between Non-Holiday and Holiday Periods:

1. **Casual Riders and Total Riders:**

- **Holidays:** The average count of casual riders and the total count of riders (casual + registered) are higher on holidays. This suggests that casual users tend to rent bikes more frequently during holidays, possibly for recreational purposes or leisure activities.

1. **Registered Riders:**

- **Holidays:** On average, registered riders rented more bikes on holidays compared to non-holiday periods. This indicates a higher demand for bike rentals among registered users during holidays, potentially due to different commuting patterns or increased outdoor activities.

In [28]:

```
# Count of data as per the Working Days

count_matrix = final_df.groupby(["workingday"])["index"].count().sort_value
s(ascending=False).reset_index()
count_matrix.columns=["Working Day (1-Working Day, 0 - Holiday)","count_of_
data_points"]
count_matrix
```

Out[28]:

	Working Day (1-Working Day, 0 - Holiday)	count_of_data_points
0	1	6958
1	0	2786

In [29]:

```
# Statistical Central Tendency analysis of data as per the working day

count_matrix = final_df.groupby(["workingday"])[["temp","atemp","humidit
y","windspeed","casual","registered","count"]].mean().reset_index()
count_matrix.columns=["Working Day (1-Working Day, 0- Holiday)","Average Te
mperature","Average Feeling Temperature","Average Humidity","Average Windsp
eed","Average Casual Bike Rents","Average Regisetered Bike Rents","Average
Count of Bike Rents"]
count_matrix
```

Out[29]:

	Working Day (1-Working Day, 0-Holiday)	Average Temperature	Average Feeling Temperature	Average Humidity	Average Windspeed	Average Casual Bike Rents	Average Regisetered Bike Rents	Ave Cou Bike R
0	0	18.101515	21.412215	65.637473	12.313655	28.110194	92.365398	120.47
1	1	20.112627	23.510688	62.278385	12.795207	22.260707	139.381000	161.64

Analysis and Explanation

After removing outliers from the dataset, we examined the distribution of data points between working days and non-working days, and analyzed the average counts of casual riders, registered riders, and total riders (casual + registered) for each category.

Distribution Between Working and Non-Working Days Post Outlier Removal:

1. **Consistent Trend in Data Distribution:**

- Similar to the initial observation, the dataset continues to show a higher count of data points on working days compared to non-working days. This reaffirms the trend that more bike rentals occur on weekdays, likely due to commuting to workplaces.

Average Counts Between Working and Non-Working Days:

1. **Casual Riders:**

- **Non-Working Days:** The average count of casual riders is higher on non-working days. This suggests that casual users prefer to rent bikes more frequently on weekends and holidays, possibly for leisure activities.

1. **Registered Riders and Total Riders:**

- **Working Days:** On average, registered riders and the total count of riders (casual + registered) rented significantly more bikes on working days compared to non-working days. This indicates a higher demand for bike rentals among registered users during weekdays, aligning with commuting patterns.

In [30]:

```
# Count of data as per the weather

count_matrix = final_df.groupby(["weather"])["index"].count().sort_values(a
scending=False).reset_index()
count_matrix.columns=["Weather","count_of_data_points"]
count_matrix
```

Out[30]:

	Weather	count_of_data_points
0	1	6314
1	2	2604
2	3	825
3	4	1

In [31]:

```
# Statistical Central Tendency analysis of data as per the weather

count_matrix = final_df.groupby(["weather"])[["temp","atemp","humidity","wi
ndspeed","casual","registered","count"]].mean().reset_index()
count_matrix.columns=["weather","Average Temperature","Average Feeling Temp
erature","Average Humidity","Average Windspeed","Average Casual Bike Rent
s","Average Regisetered Bike Rents","Average Count of Bike Rents"]
count_matrix
```

Out[31]:

	weather	Average Temperature	Average Feeling Temperature	Average Humidity	Average Windspeed	Average Casual Bike Rents	Average Regisetered Bike Rents	Avei Coui Bike R
0	1	19.751558	23.133510	58.065410	12.697171	25.917960	131.556383	157.47
1	2	19.108771	22.528318	69.869432	12.126249	22.528418	124.436636	146.96
2	3	19.267515	22.426321	81.876364	14.039034	13.198788	87.642424	100.84
3	4	8.200000	11.365000	86.000000	6.003200	6.000000	158.000000	164.00

Analysis and Explanation

Following the removal of outliers from the dataset, we examined the distribution of records across different weather conditions and analyzed the average counts of casual riders, registered riders, and total riders (casual + registered) for each weather category.

Distribution Across Weather Conditions Post Outlier Removal:

1. **Consistent Trend in Data Distribution:**

- Post outlier removal, each weather condition continues to maintain a similar count of records as before. Clear and cloudy weather still dominate the dataset, with the least number of data points recorded during heavy rainy weather. This suggests a consistent pattern in data distribution across different weather conditions.

Average Counts Across Weather Conditions:

1. **Casual Riders, Registered Riders, and Total Riders:**

- **Clear and Cloudy Weather:** On average, the count of casual riders, registered riders, and total riders (casual + registered) is higher during clear and cloudy weather conditions. This indicates a preference among users for renting bikes during favorable weather, possibly for commuting or outdoor activities.

1. **Mist and Cloudy Weather:** Following clear and cloudy weather, mist and cloudy conditions show the next highest averages for casual riders, registered riders, and total riders. This suggests that users are still willing to use bike rentals under slightly less favorable weather conditions.

In [32]:

```
# Count of data as per the Hour of the day

count_matrix = final_df.groupby(["hour_of_the_day"])["index"].count().sort_values(ascending=False).reset_index()
count_matrix.columns=["hour_of_the_day","count_of_data_points"]
count_matrix
```

Out[32]:

	hour_of_the_day	count_of_data_points
0	23	456
1	6	455
2	22	455
3	0	455
4	1	454
5	21	454
6	9	453
7	5	452
8	2	448
9	20	446
10	7	445
11	4	442
12	3	433
13	10	415
14	19	396
15	11	388
16	12	375
17	16	371
18	13	366
19	15	363
20	14	356
21	8	331
22	18	290
23	17	245

In [33]:

```
# Statistical Central Tendency analysis of data as per the hour of the day

count_matrix = final_df.groupby(["hour_of_the_day"])[["temp","atemp","humid
ity","windspeed","casual","registered","count"]].mean().reset_index()
count_matrix.columns=["hour_of_the_day","Average Temperature","Average Feel
ing Temperature","Average Humidity","Average Windspeed","Average Casual Bik
e Rents","Average Regisetered Bike Rents","Average Count of Bike Rents"]
count_matrix
```

Out[33]:

	hour_of_the_day	Average Temperature	Average Feeling Temperature	Average Humidity	Average Windspeed	Average Casual Bike Rents	Averag Registere Bike Rent
0	0	19.013187	22.462582	68.079121	10.701564	10.312088	44.82637
1	1	18.639648	22.011476	69.581498	10.418839	6.513216	27.34581
2	2	18.455491	21.822623	70.622768	10.125315	4.819196	18.08035
3	3	18.433903	21.814007	72.293303	10.173416	2.681293	9.07621
4	4	18.036290	21.352738	73.640271	10.717605	1.262443	5.14479
5	5	17.610044	20.882002	73.409292	10.062407	1.455752	18.31194
6	6	17.481319	20.722747	73.934066	10.433402	4.149451	72.10989
7	7	17.713843	20.956798	72.262921	10.928242	10.721348	194.89662
8	8	17.613897	20.887251	69.746224	12.075829	19.202417	230.72507
9	9	19.325210	22.712296	65.441501	12.945125	30.492274	190.86975
10	10	19.794602	23.081386	61.009639	14.055615	36.631325	116.83132
11	11	20.307680	23.617539	57.059278	14.371465	41.170103	128.89690
12	12	21.007307	24.383827	53.730667	14.938945	43.338667	163.95200
13	13	21.651585	25.062008	50.819672	15.574077	43.612022	158.20491
14	14	22.232135	25.751180	49.348315	15.817646	43.786517	138.25000
15	15	22.338788	25.863898	49.035813	15.793524	45.471074	154.31129
16	16	22.184205	25.577722	49.547170	16.334437	47.778976	230.09164
17	17	19.345306	22.566327	53.620408	16.155177	39.791837	280.61632
18	18	19.450966	22.721897	55.179310	15.645078	37.375862	275.56551
19	19	20.237020	23.591995	57.075758	14.298240	38.603535	246.41666
20	20	20.534933	24.047119	59.670404	13.080849	34.336323	191.04484
21	21	20.164053	23.661762	62.429515	12.055233	27.775330	144.06607
22	22	19.735868	23.196857	64.586813	11.837781	22.268132	110.49890
23	23	19.343728	22.775548	66.649123	11.077304	15.462719	74.04605

Analysis and Explanation

After removing outliers from the dataset, we analyzed the distribution of data points across different hours of the day and examined the average counts of casual riders, registered riders, and total riders (casual + registered) for each hour.

Distribution Across Hours Post Outlier Removal:

1. **Hourly Distribution:**

- Post outlier removal, the dataset shows a relatively consistent count of data points from midnight to 1 PM, with a gradual decrease thereafter until midnight. This distribution suggests a standard sampling across different hours of the day.

Average Counts Across Hours:

1. **Casual Riders:**

- Casual rider counts, on average, peak between 11 AM to 4 PM, with more than 40 bikes rented per hour during this period. This indicates higher usage of bikes by casual users during midday hours, possibly for leisure or recreational purposes.

1. **Registered Riders:**

- Registered rider counts, on average, show distinct patterns: Morning Hours (7 AM to 9 AM): **There is an increase in bike rentals, reflecting the start of office hours and commuting patterns. Afternoon (3 PM):** Rentals decrease during mid-afternoon. Evening Surge (4 PM to 7 PM):** There is a significant surge in bike rentals, peaking at 5 PM with an average of 280 bikes rented. This period likely corresponds to the end of office hours and commuting back home.

1. **Total Count of Rented Bikes:**

- The total count of rented bikes, on average, mirrors the trend observed in registered rider counts. This correlation suggests that registered riders largely influence the overall bike rental patterns throughout the day.

In [34]:

```
# Count of data as per the Month

count_matrix = final_df.groupby(["month"])["index"].count().reset_index()
count_matrix.columns=["Month","count_of_data_points"]
count_matrix
```

Out[34]:

	Month	count_of_data_points
0	1	879
1	2	886
2	3	835
3	4	794
4	5	797
5	6	755
6	7	763
7	8	794
8	9	748
9	10	785
10	11	837
11	12	871

In [35]:

```
# Statistical Central Tendency analysis of data as per the month

count_matrix = final_df.groupby(["month"])[["temp","atemp","humidity","windspeed","casual","registered","count"]].mean().reset_index()
count_matrix.columns=["Month","Average Temperature","Average Feeling Temperature","Average Humidity","Average Windspeed","Average Casual Bike Rents","Average Regisetered Bike Rents","Average Count of Bike Rents"]
count_matrix
```

Out[35]:

	Month	Average Temperature	Average Feeling Temperature	Average Humidity	Average Windspeed	Average Casual Bike Rents	Average Regisetered Bike Rents	Aver Coun Bike Re
0	1	9.778430	12.007520	55.986348	14.562385	7.465301	81.055745	88.521
1	2	11.720632	14.439193	56.391648	13.967609	9.205418	94.590293	103.795
2	3	15.476886	18.526228	57.051497	15.300906	18.823952	99.380838	118.204
3	4	18.291990	21.827941	58.280856	15.224383	26.638539	107.569270	134.207
4	5	22.204818	26.111255	70.392723	12.199607	30.722710	139.766625	170.489
5	6	26.711364	30.688801	61.340397	12.269129	33.855629	145.188079	179.043
6	7	30.432425	34.537202	60.558322	10.719684	38.934469	145.849279	184.783
7	8	29.411562	32.844402	64.510076	11.458730	37.568010	145.788413	183.356
8	9	25.236925	28.930348	73.697861	11.216955	29.094920	134.486631	163.581
9	10	20.463439	24.214414	72.123567	10.825792	26.080255	143.740127	169.820
10	11	15.036272	18.151458	61.966547	13.081594	20.351254	142.948626	163.299
11	12	13.759242	17.039311	68.495982	10.554977	14.655568	140.717566	155.373

Analysis and Explanation

After removing outliers from the dataset, we examined the distribution of data points across different months and analyzed the average counts of casual riders, registered riders, and total riders (casual + registered) for each month.

Distribution Across Months Post Outlier Removal:

1. **Monthly Distribution:**

- Post outlier removal, the dataset shows a relatively similar count of data points from January to March, with a gradual decrease observed from April to November. This indicates a consistent sampling across the early months followed by a decline towards the end of the year.

Average Counts Across Months:

1. **Casual Riders:**

Casual rider counts, on average, peak from April to July, with more than 30 bikes rented per month during this period. The highest average is observed in June, suggesting increased bike usage by casual users during summer months, possibly for recreational purposes.

1. **Registered Riders:**

- Registered rider counts, on average, show a broader distribution: March to December:** There is consistent bike rental activity, with more than 100 bikes rented per month on average. Peak months for registered ridership are June and July, indicating heightened demand during the summer months, possibly influenced by weather conditions and commuting patterns.

1. **Total Count of Rented Bikes:**

- The total count of rented bikes, on average, aligns closely with the trend observed in registered rider counts. This indicates that registered riders significantly impact the overall bike rental patterns across different months.

In [36]:

```
# Count of data as per the Quarter

count_matrix = final_df.groupby(["quarter"])["index"].count().sort_values(ascending=False).reset_index()
count_matrix.columns=["Quarter","count_of_data_points"]
count_matrix
```

Out[36]:

	Quarter	count_of_data_points
0	1	2600
1	4	2493
2	2	2346
3	3	2305

In [37]:
```
# Statistical Central Tendency analysis of data as per the quarter

count_matrix = final_df.groupby(["quarter"])[["temp","atemp","humidity","wi
ndspeed","casual","registered","count"]].mean().reset_index()
count_matrix.columns=["Quarter","Average Temperature","Average Feeling Temp
erature","Average Humidity","Average Windspeed","Average Casual Bike Rent
s","Average Regisetered Bike Rents","Average Count of Bike Rents"]
count_matrix
```

Out[37]:

	Quarter	Average Temperature	Average Feeling Temperature	Average Humidity	Average Windspeed	Average Casual Bike Rents	Average Regisetered Bike Rents	Aver Cour Bike Re
0	1	12.270354	14.929667	56.466538	14.596883	11.706154	91.553077	103.259
1	2	22.330844	26.134740	63.380222	13.245712	30.348679	130.614237	160.962
2	3	28.394768	32.134594	66.183514	11.135633	35.270716	142.140998	177.411
3	4	16.299021	19.672012	67.446049	11.488538	20.165263	142.418371	162.583

Analysis and Explanation

After removing outliers from the dataset, we examined the distribution of records across different quarters and analyzed the average counts of casual riders, registered riders, and total riders (casual + registered) for each quarter.

Distribution Across Quarters Post Outlier Removal:

1. **Quarterly Distribution:**

- Post outlier removal, each quarter shows a relatively similar count of records, with a slight increase observed in the first quarter compared to other quarters. This suggests a balanced distribution of data across the four quarters of the year.

Average Counts Across Quarters:

1. **Casual Riders and Total Riders:**

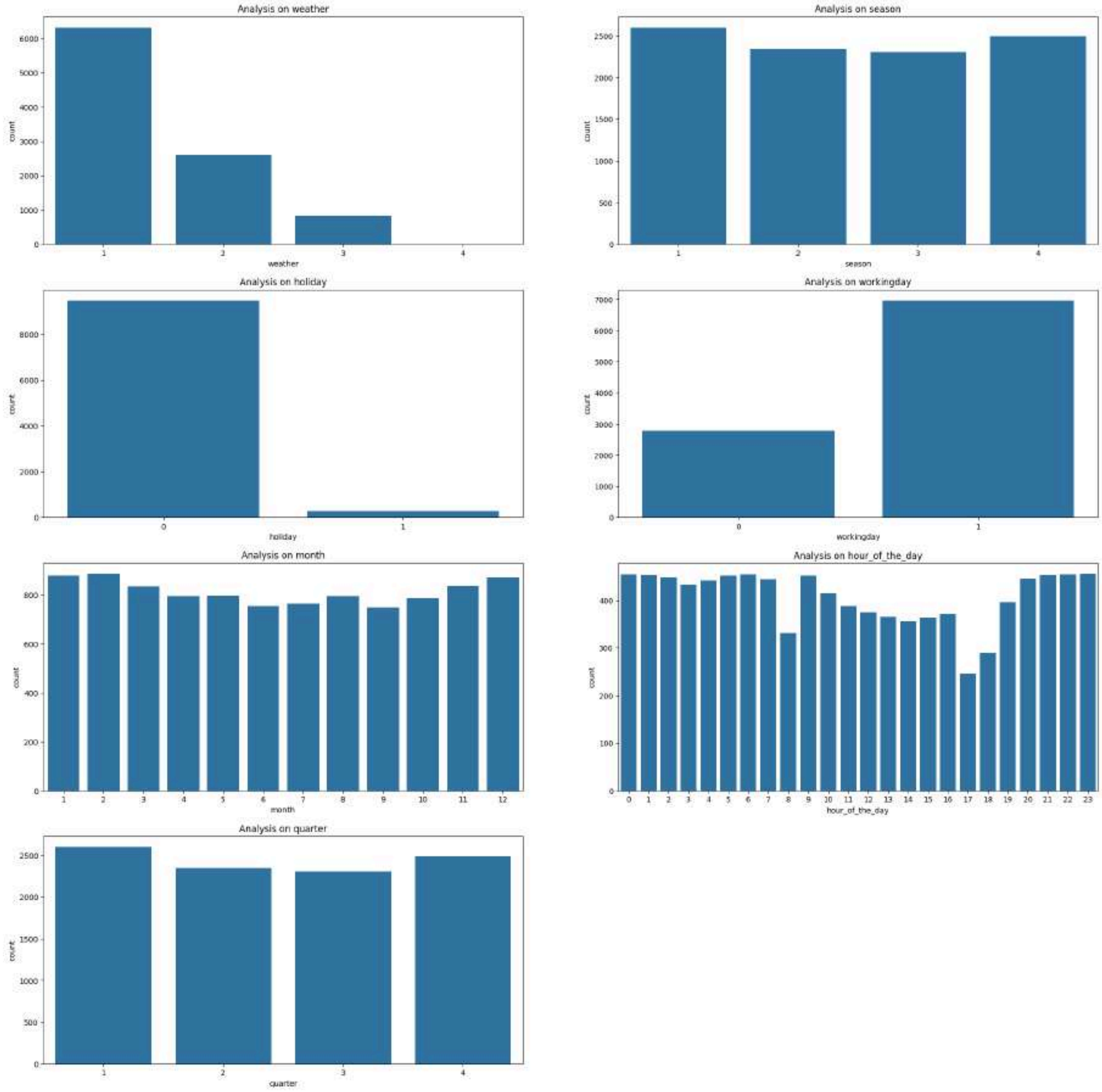
- **Third Quarter:** On average, the count of casual riders and the total count of riders (casual + registered) are higher during the third quarter. This indicates that casual users tend to rent bikes more frequently during the summer months (July to September).

1. **Registered Riders:**

- **Fourth Quarter:** On average, registered riders rented slightly more bikes during the fourth quarter compared to the third quarter. This suggests a preference among registered users for bike rentals during the fall and early winter months (October to December), possibly due to commuting patterns and seasonal changes.

Visual Analysis - Univariate & Bivariate Analysis

In [38]:
```
# Analysis on the features provided with their corresponding count in the d
atapoints
plt.figure(figsize = (25,25))
features = ['weather', 'season', 'holiday', 'workingday',"month","hour_of_t
he_day","quarter"]
for i in range(len(features)):
     plt.subplot(4, 2, i+1)
     sns.countplot(x = final_df[features[i]])
     plt.title('Analysis on {}'.format(features[i]))

plt.show()
```



Analysis and Explanation

Visual analysis complements our non-visual findings, providing further insights into the trends observed across different variables:

Weather Conditions:

- Clear and Cloudy Weather: This weather condition exhibits the highest number of data points, indicating its prevalence in the dataset. Users tend to rent bikes more frequently during clear or partly cloudy conditions.

Seasons:

- Spring and Winter: These seasons show a similar number of data points, suggesting a balanced representation in the dataset. The trends observed across seasons align with expected seasonal patterns in bike rental behavior.

Day Types:

- Working Days and Non-Holidays: These periods have the highest number of data points, reflecting increased bike rental activity during regular weekdays and weekends when users are likely commuting or engaging in leisure activities.

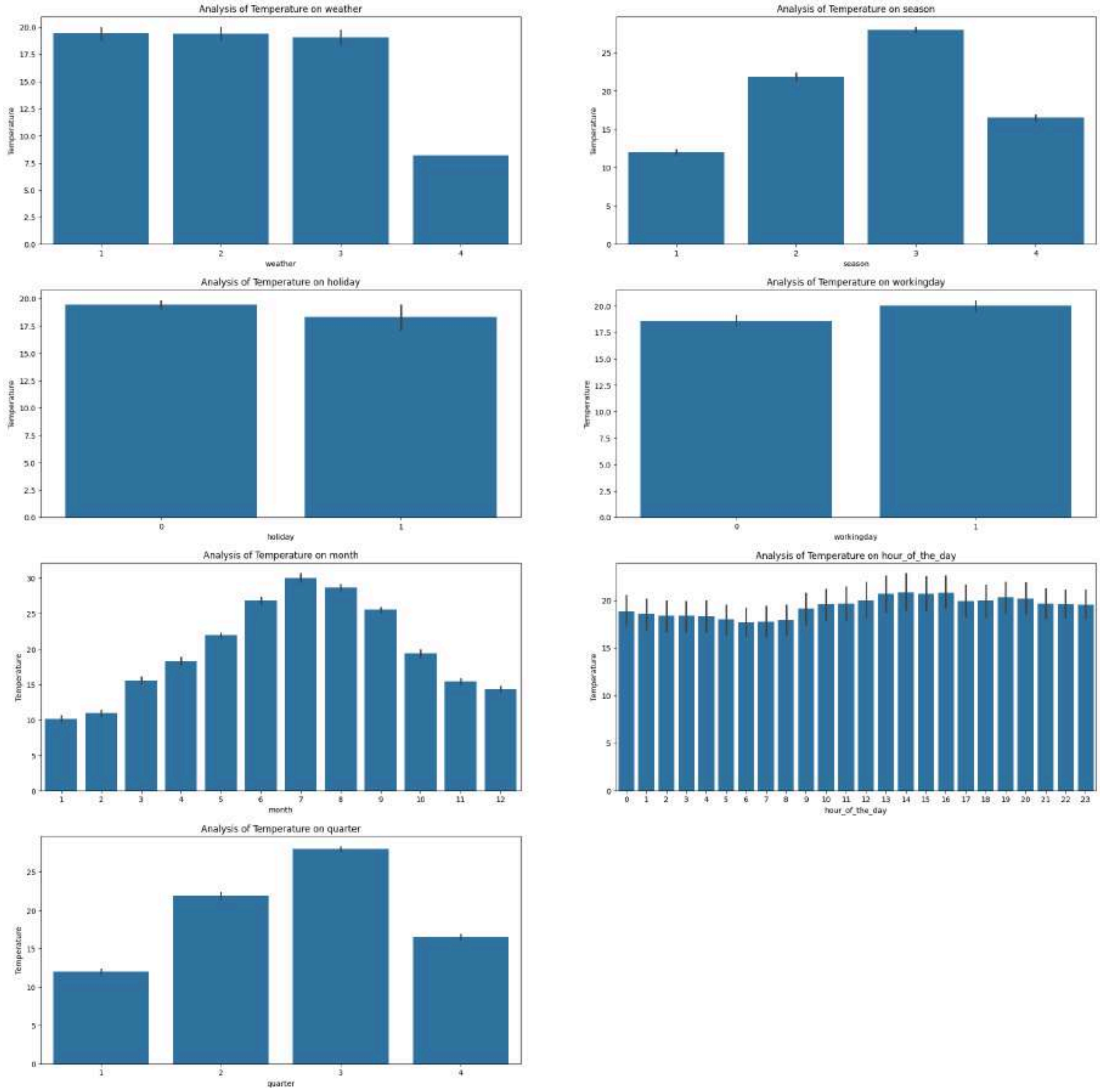
Months:

- Winter and Fall: These months contain the majority of data points, indicating higher bike rental frequency during colder months. This trend correlates with seasonal preferences and weather conditions.

Time of Day:

- Night vs. Noon: The dataset shows a majority of data points during nighttime hours, with fewer observations during the afternoon. This suggests that bike rentals are more frequent during morning and evening hours, potentially aligning with commuting times.

```
In [39]: # Plotting average temperature for all the features

         plt.figure(figsize = (25,25))
         features = ['weather', 'season', 'holiday', 'workingday',"month","hour_of_the_day","quarter"]
         count_df =  final_df.groupby(features)["temp"].mean().reset_index()
         for i in range(len(features)):
                  plt.subplot(4, 2, i+1)
                  sns.barplot(x = count_df[features[i]], y = count_df["temp"])
                  plt.ylabel('Temperature')
                  plt.title('Analysis of Temperature on {}'.format(features[i]))
         plt.show()
```



Analysis and Explanation

The average temperature analysis provides insights into how weather conditions, seasons, day types, months, and time of day correlate with bike rental patterns:

Weather Conditions:

- Temperature by Weather: The highest average temperatures are observed during clear and cloudy weather conditions, followed by mist and cloudy weather. This indicates that warmer weather tends to correlate with higher bike rental activity, likely due to favorable outdoor conditions.

Seasons and Quarters:

- Temperature by Season and Quarter: Fall Season and Third Quarter: These periods show higher average temperatures compared to summers or the second quarter. This observation, particularly the increase in the third quarter, may be influenced by seasonal changes and the removal of outliers affecting temperature data.

Day Types:

- Temperature by Day Type: Working Days and Non-Holidays: Average temperatures are higher on these days, suggesting that users are more likely to rent bikes when the weather is warmer and conducive to outdoor activities.

Months:

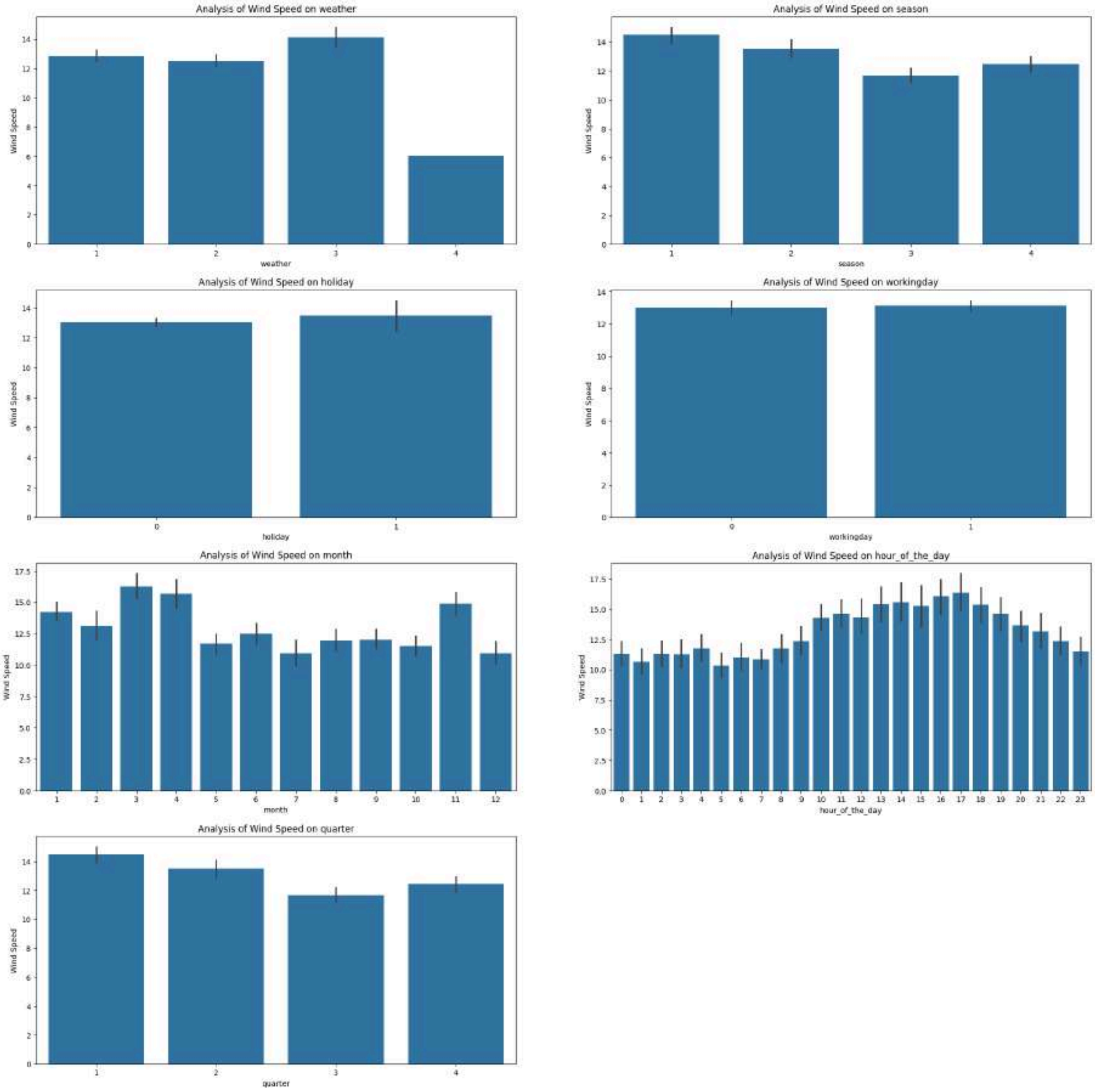
- Temperature by Month: March to November: Average temperatures gradually rise from March to November. This trend aligns with seasonal changes, indicating peak bike rental periods during warmer months when temperatures are more favorable for outdoor activities.

Time of Day:

- Temperature Throughout the Day: Average temperatures remain relatively consistent throughout the day, with slightly higher temperatures observed during the afternoon. This suggests that bike rental patterns are influenced by favorable temperature conditions during the day, with peak activity likely during warmer hours.

In [40]:

```python
# Plotting average wind speed for all the features

plt.figure(figsize = (25,25))
features = ['weather', 'season', 'holiday', 'workingday',"month","hour_of_the_day","quarter"]
count_df =  final_df.groupby(features)["windspeed"].mean().reset_index()
for i in range(len(features)):
        plt.subplot(4, 2, i+1)
        sns.barplot(x = count_df[features[i]], y = count_df["windspeed"])
        plt.ylabel('Wind Speed')
        plt.title('Analysis of Wind Speed on {}'.format(features[i]))
plt.show()
```



Analysis and Explanation

The average windspeed analysis provides insights into how weather conditions, seasons, day types, months, and time of day correlate with bike rental patterns:

Weather Conditions:

- **Windspeed by Weather:** The highest average windspeeds are observed during light snow and light rainy weather conditions, followed by mist and cloudy weather. This suggests that certain weather conditions conducive to precipitation are associated with higher windspeeds, potentially impacting bike rental decisions due to safety concerns or discomfort.

Seasons and Quarters:

- Windspeed by Season and Quarter: ** Spring Season and First Quarter: These periods show higher average windspeeds compared to summers or the second quarter. This observation aligns with seasonal weather patterns where spring often experiences higher winds. The first quarter's data showing higher windspeeds may reflect specific weather patterns affecting that period.

Day Types:

- Windspeed by Day Type: ** Similar on Working Days and Non-Holidays: The average windspeed remains consistent across both working days and non-holidays. This suggests that windspeed does not significantly vary based on these day types, indicating similar riding conditions throughout the week.

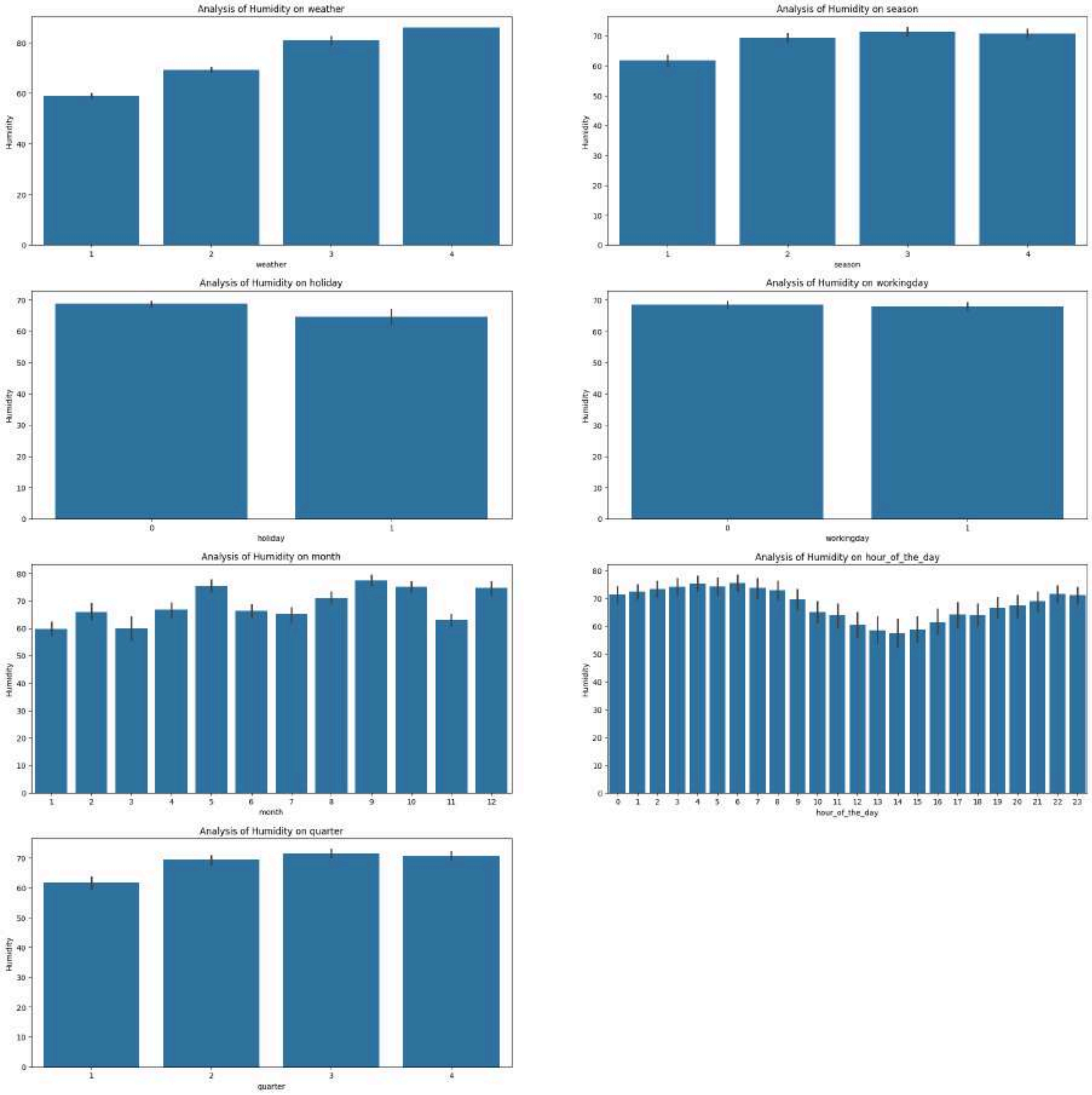
Months:

- Windspeed by Month: ** March, April, and November: These months exhibit higher average windspeeds. This trend may correspond to transitional weather periods, such as spring and late fall, when wind patterns can be more turbulent.

Time of Day:

- Windspeed Throughout the Day: ** Average windspeed increases from 10 AM to 5 PM and then decreases towards midnight. This pattern indicates higher winds during daytime hours, potentially influencing bike rental decisions due to the perceived difficulty of riding in windy conditions.

```
In [41]: # Plotting average humidity for all the features

         plt.figure(figsize = (25,25))
         features = ['weather', 'season', 'holiday', 'workingday',"month","hour_of_t
         he_day","quarter"]
         count_df =  final_df.groupby(features)["humidity"].mean().reset_index()
         for i in range(len(features)):
                  plt.subplot(4, 2, i+1)
                  sns.barplot(x = count_df[features[i]], y = count_df["humidity"])
                  plt.ylabel('Humidity')
                  plt.title('Analysis of Humidity on {}'.format(features[i]))
         plt.show()
```



Analysis and Explanation

The average humidity analysis provides insights into how weather conditions, seasons, day types, months, and time of day correlate with bike rental patterns:

Weather Conditions:

- Humidity by Weather: The highest average humidity levels are observed during heavy rainy weather conditions, followed by light snow and light rainy weather. This suggests that humid weather conditions associated with precipitation influence bike rental decisions, possibly due to discomfort or safety concerns.

Seasons and Quarters:

- Humidity by Season and Quarter: ** Fall Season and Third Quarter: These periods show higher average humidity levels compared to summers or the second quarter, and winter or fourth quarter. This observation aligns with seasonal weather patterns where fall often experiences higher humidity levels. The third quarter's data showing higher humidity may reflect specific weather patterns affecting that period.

Day Types:

- Humidity by Day Type: ** Similar on Working Days and Non-Holidays: The average humidity remains consistent across both working days and non-holidays. This suggests that humidity does not significantly vary based on these day types, indicating similar riding conditions throughout the week.

Months:

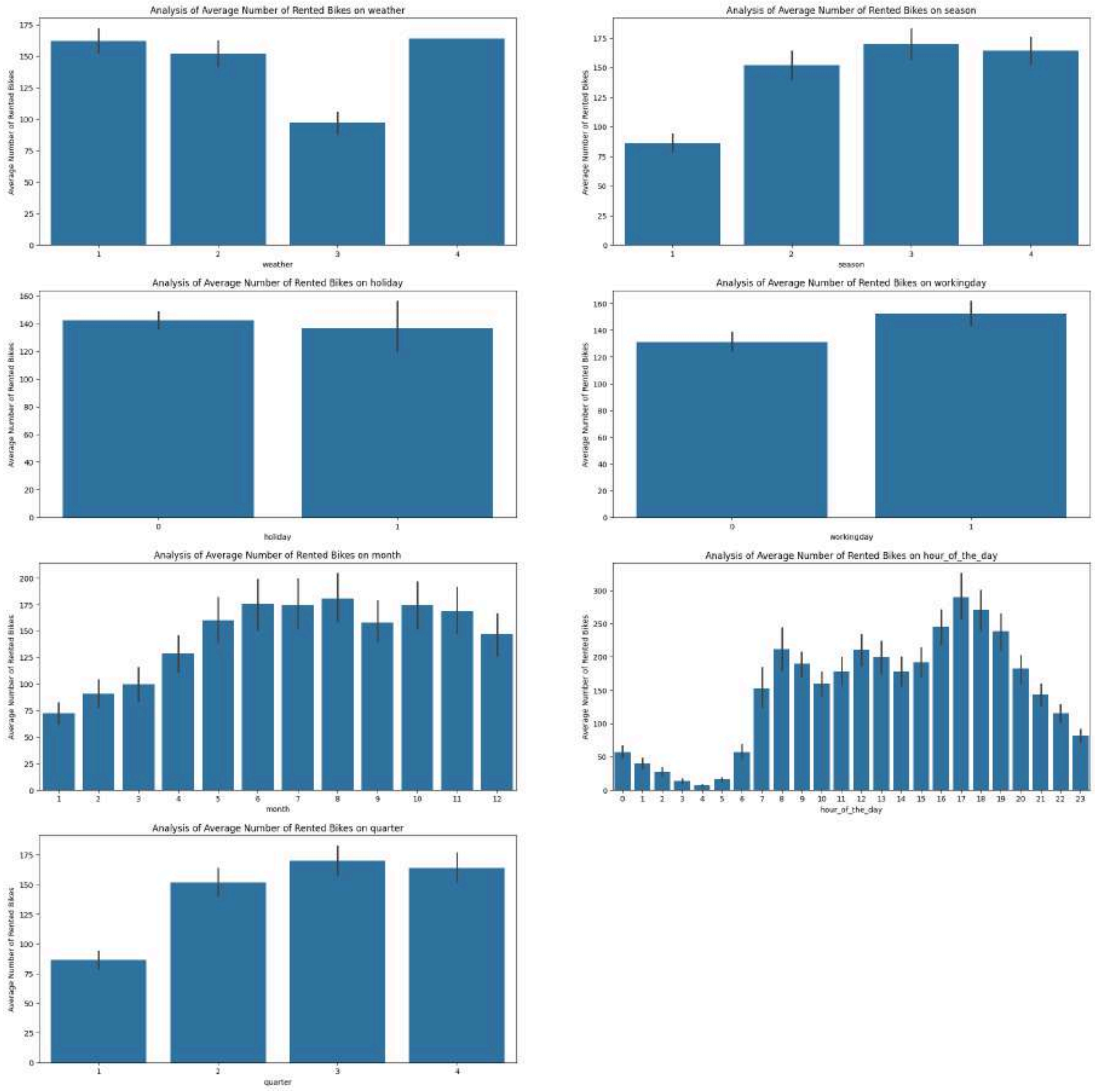
- Humidity by Month: ** May, September, October, and December: These months exhibit higher average humidity levels. This trend may correspond to transitional weather periods or seasonal changes, where humidity levels tend to be higher.

Time of Day:

- Humidity Throughout the Day: ** Average humidity levels remain consistently higher from 12 midnight to 8 AM, gradually decreasing until 2 PM. After 2 PM, humidity levels start slowly increasing again until midnight. This diurnal pattern indicates higher humidity levels during early morning and nighttime hours, with a decrease during midday, possibly influenced by temperature fluctuations and atmospheric conditions.

In [42]:

```
# Plotting average count of rented bikes against all features

plt.figure(figsize = (25,25))
features = ['weather', 'season', 'holiday', 'workingday',"month","hour_of_t
he_day","quarter"]
count_df =  final_df.groupby(features)["count"].mean().reset_index()
for i in range(len(features)):
        plt.subplot(4, 2, i+1)
        sns.barplot(x = count_df[features[i]], y = count_df["count"])
        plt.ylabel('Average Number of Rented Bikes')
        plt.title('Analysis of Average Number of Rented Bikes on {}'.forma
t(features[i]))
plt.show()
```



Analysis and Explanation

The average count of rented bikes analysis provides insights into how weather conditions, seasons, day types, months, and time of day correlate with bike rental patterns:

Weather Conditions:

- Rental Count by Weather: The highest average count of rented bikes is observed during clear and cloudy weather conditions, followed by mist and cloudy weather. The higher count observed during heavy rain weather is likely skewed due to a single data point present for that weather condition, which may not represent typical usage patterns.

Seasons and Quarters:

- Rental Count by Season and Quarter: ** Fall Season and Third Quarter: These periods show higher average counts of rented bikes compared to summers or the second quarter, and winter or fourth quarter. This observation aligns with seasonal preferences where fall, particularly the third quarter, sees increased bike rental activity possibly due to favorable weather conditions.

Day Types:

- Rental Count by Day Type: ** Slightly More on Working Days and Non-Holidays: The average count of rented bikes shows a slight preference on working days and non-holidays. This indicates consistent rental demand throughout the week, with potentially higher usage for commuting and leisure activities on these days.

Months:

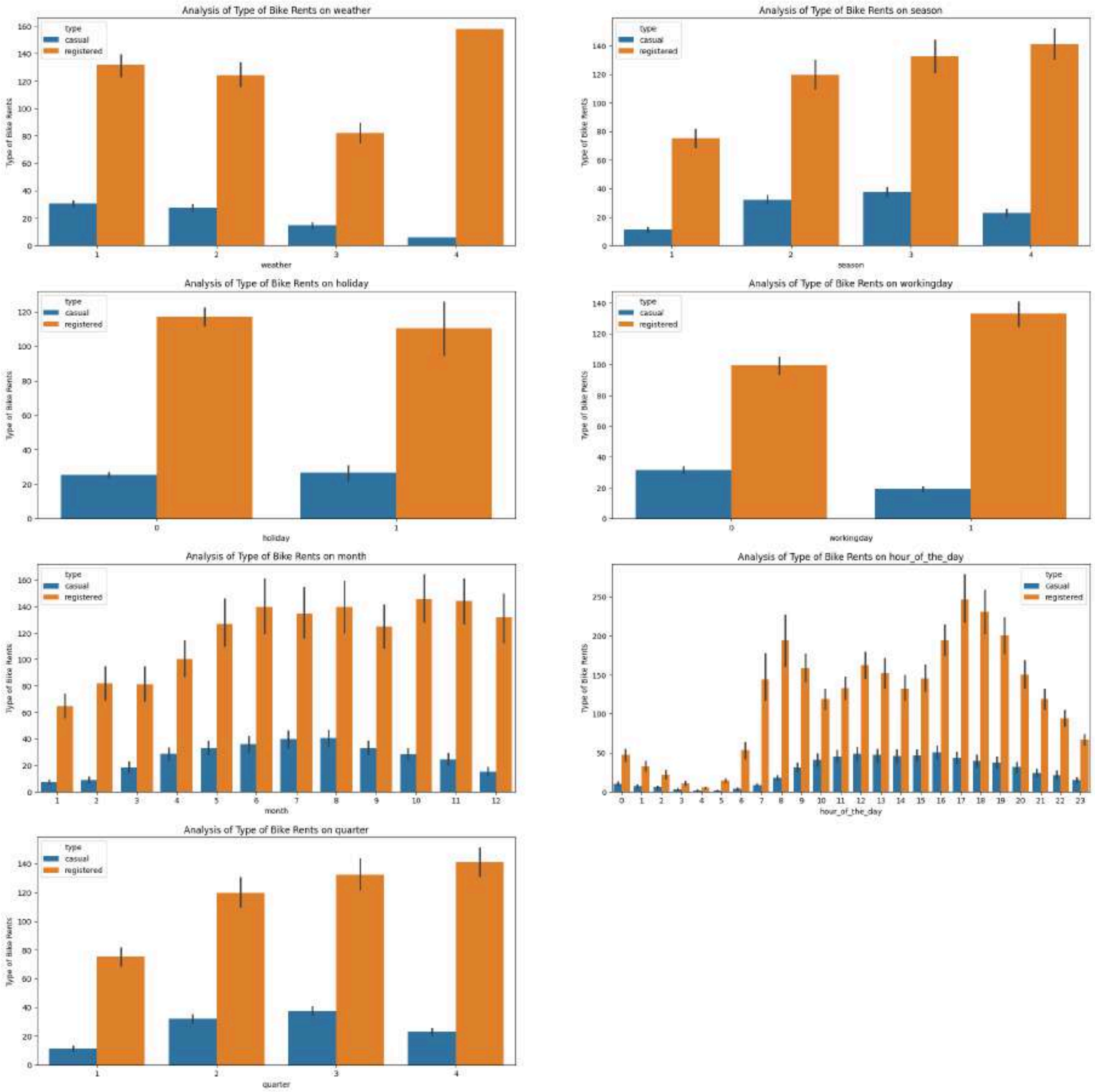
- Rental Count by Month: ** January to August, October: The average count of rented bikes starts rising from January and peaks in August, followed by another peak in October. There is a decrease observed from October to December. This trend correlates with seasonal variations and weather conditions favorable for outdoor activities.

Time of Day:

- Rental Count Throughout the Day: ** The average count of rented bikes remains consistently higher from 7 AM to 9 AM, likely corresponding to morning commute hours. It then decreases until 3 PM and begins to rise again from 4 PM to 7 PM, peaking at 5 PM. This pattern reflects typical commuting patterns and recreational usage during after-work hours.

In [43]:

```
# Plotting average count of registered and casual riders renting bikes against all features

plt.figure(figsize = (25,25))
features = ['weather', 'season', 'holiday', 'workingday',"month","hour_of_the_day","quarter"]
count_df =  final_df.groupby(features)[["casual","registered"]].mean().reset_index()
count_df = pd.melt(count_df,id_vars = features)
count_df.columns = ['weather', 'season', 'holiday', 'workingday',"month","hour_of_the_day","quarter","type","mean"]
for i in range(len(features)):
        plt.subplot(4, 2, i+1)
        sns.barplot(x = count_df[features[i]], y = count_df["mean"],hue= count_df["type"],color = )
        plt.ylabel('Type of Bike Rents')
        plt.title('Analysis of Type of Bike Rents on {}'.format(features[i]))
plt.show()
```



Analysis and Explanation

The analysis focuses on the average count of registered and casual riders renting bikes across various factors:

Rider Type Comparison:

- Registered vs. Casual Riders: ** The average count of registered riders renting bikes is significantly higher than that of casual riders. This indicates that registered users form the majority of bike rental customers, likely due to regular and frequent usage patterns.

Weather Conditions:

- Rental Count by Weather: ** Both registered and casual riders show higher average counts of bike rentals during clear and cloudy weather conditions, followed by mist and cloudy weather. It’s important to note that the higher count observed during heavy rain weather is skewed by a single data point, which may not reflect typical usage patterns.

Seasons and Quarters:

- Registered vs. Casual Riders by Season and Quarter: ** Registered Riders: Prefer winter season or the fourth quarter, likely due to cooler temperatures and potentially fewer weather disruptions. Casual Riders: Prefer fall season or the third quarter, with a secondary preference for summers or the second quarter. This trend suggests that casual riders may favor milder weather conditions for bike rentals.

Day Types:

- Registered vs. Casual Riders by Day Type: ** Registered Riders: Show slightly higher average bike rentals on working days and non-holidays. This aligns with commuting and regular usage patterns associated with registered users. Casual Riders: Rent bikes more on holidays and non-working days, indicating leisure or occasional usage patterns among casual riders.

Months:

- Registered vs. Casual Riders by Month: ** Both registered and casual riders see a rise in bike rentals from April to August. However, casual ridership decreases after August, whereas registered ridership increases again in October before tapering off towards December. This pattern reflects seasonal preferences and weather impacts on bike rental behavior.

Time of Day:

- Registered vs. Casual Riders Throughout the Day: ** Both groups show a similar trend in bike rental counts throughout the day. Rentals increase from 7 AM to 9 AM, decline until 3 PM, and then rise again from 4 PM to 7 PM, peaking at 5 PM. This pattern corresponds to typical commuting and leisure usage patterns, where bike rentals are highest during morning and late afternoon hours.

Hypothesis Testing

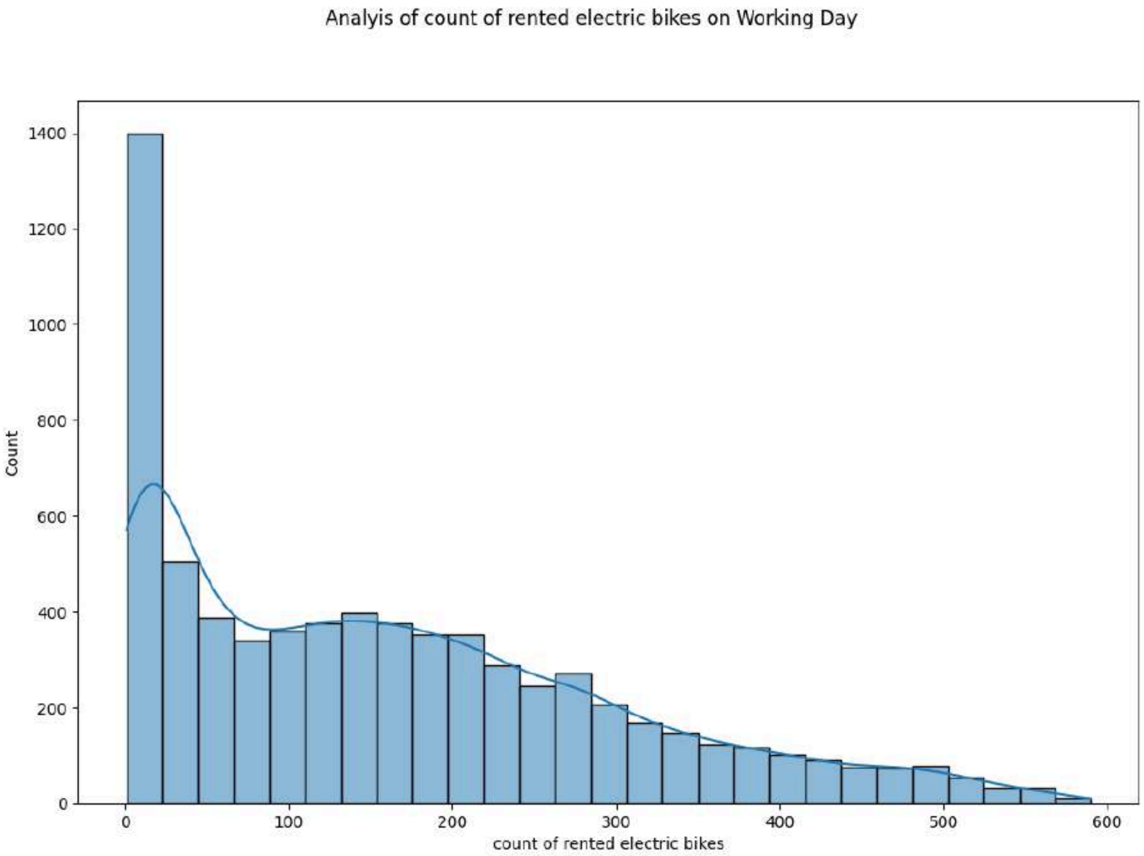
1. Whether Working Day has an effect on the count of rented electric bikes

In [44]:

```
# Creating two different dataframes for working day count of renting bikes as working_day_df and non-working days count of renting bikes as holiday_df

working_day_df = final_df.loc[(final_df["workingday"]==True),"count"]
holiday_df = final_df.loc[(final_df["workingday"]==False),"count"]
```

In [45]:

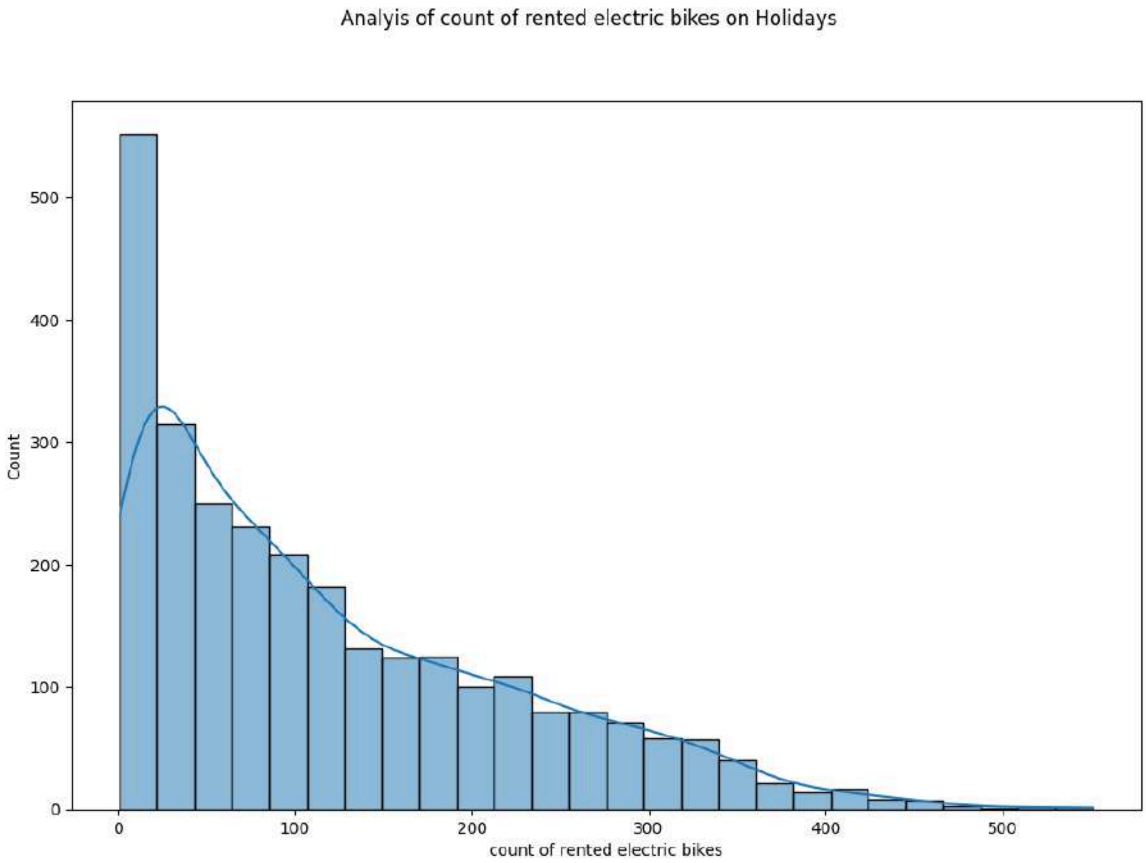
```
# checking for the dataset's normal distribution with Wilkin-Shapiro Test

# Null Hypothesis(H0) : Data is Normally distributed
# Alternate Hypothesis(Ha) : Data is not Normally distributed

plt.figure(figsize = (12,8))
sns.histplot(working_day_df,kde=True)
plt.suptitle("Analyis of count of rented electric bikes on Working Day")
plt.xlabel("count of rented electric bikes")
plt.show()

working_day_sample_df = working_day_df.sample(100)
shapiro_stat,p_value = shapiro(working_day_sample_df)
alpha = 0.05
print("Confidence Interval: 95%")
print("Wilkin-Shapiro test with Test Statistic: {}, and p-value: {}".format
(shapiro_stat,p_value))
if p_value < alpha:
    print("Reject Ho: Data is not Normally distributed")
else:
    print("Failed to reject Ho: Data is Normally distributed")
```



```
Confidence Interval: 95%
Wilkin-Shapiro test with Test Statistic: 0.9320266246795654, and p-value:
6.497644790215418e-05
Reject Ho: Data is not Normally distributed
```

In [46]:

```
# checking for the dataset's normal distribution with Wilkin-Shapiro Test

# Null Hypothesis(H0) : Data is Normally distributed
# Alternate Hypothesis(Ha) : Data is not Normally distributed

plt.figure(figsize = (12,8))
sns.histplot(holiday_df,kde=True)
plt.suptitle("Analyis of count of rented electric bikes on Holidays")
plt.xlabel("count of rented electric bikes")
plt.show()

holiday_sample_df = holiday_df.sample(100)
shapiro_stat,p_value = shapiro(holiday_sample_df)
alpha = 0.05
print("Confidence Interval: 95%")
print("Wilkin-Shapiro test with Test Statistic: {}, and p-value: {}".format
(shapiro_stat,p_value))
if p_value < alpha:
    print("Reject Ho: Data is not Normally distributed")
else:
    print("Failed to reject Ho: Data is Normally distributed")
```



```
Confidence Interval: 95%
Wilkin-Shapiro test with Test Statistic: 0.8988927602767944, and p-value:
1.254558696928143e-06
Reject Ho: Data is not Normally distributed
```

In [47]:

```
# Check for Equal or Similar variance among the two groups

print("Variance for Working Day group is : {}\nVariance for Non-Working Day group is : {}".format(np.var(working_day_df),np.var(holiday_df)))
print("The Ratio of the above two is : {}".format(np.var(working_day_df)/np.var(holiday_df)))
```

```
Variance for Working Day group is : 19139.340295561404
Variance for Non-Working Day group is : 11308.06203886317
The Ratio of the above two is : 1.6925393785233895
```

Setting up Null and Alternate Hypothesis

Null Hypothesis (H0): The count of rented electric cycles is the same on working days and holidays.

Alternate Hypothesis (Ha): The count of rented electric cycles is higher on working days than on holidays.

Significance Level (α): 0.05

Justification for T-Test:

- Although the data is not normally distributed, we are comparing two categories: working days (categorical) and count of rented electric cycles (numerical).
- The variance ratio between the two groups is low, which supports the use of the Two Sample Independent T-Test for comparing means.

Assumptions and Considerations:

- Data Distribution: While the count of rented electric cycles may not follow a normal distribution, the t-test is robust to deviations from normality, especially with larger sample sizes.
- Equal Variances: The assumption of equal variances between working days and holidays should ideally hold, but the t-test can still be applied with caution if variances are unequal.

Interpretation:

- If the p-value from the t-test is less than 0.05, we reject the null hypothesis. *If the p-value is greater than or equal to 0.05, we fail to reject the null hypothesis.

In [48]:

```python
# Implementing Two Sample Independent T-Test

test_statistic,p_value = ttest_ind(working_day_df,holiday_df,alternative="greater")
print("Two Sample Independent T-Test with Test Statistic: {}, and p-value: {}".format(test_statistic,p_value))
alpha = 0.05
print("Confidence Interval: 95%")
if p_value < alpha:
    print("Reject Ho: Working Day has an impact on the count of rented electric cycles,i.e., count of rented electric bikes is more on working day than on holidays.")
else:
    print("Failed to reject Ho: Working Day doesn't have impact on the count of rented electric cycles,i.e., count of rented electric bikes is same as on working day than on holidays.")
```

```
Two Sample Independent T-Test with Test Statistic: 14.122552537877892, and p-value: 3.8137606881498196e-45
Confidence Interval: 95%
Reject Ho: Working Day has an impact on the count of rented electric cycles,i.e., count of rented electric bikes is more on working day than on holidays.
```

Analysis and Explanation

1. **Dataset Preparation:**

- Two datasets were created from final_df: one for working days and another for holidays (non-working days).

1. **Normality Check:**

- Shapiro-Wilk Test: Normality in the distribution of the data was assessed using the Shapiro-Wilk test. Unfortunately, the data did not adhere to a normal distribution.

1. **Null and Alternative Hypotheses:**

- **Null Hypothesis (H0):** The count of rented electric cycles is the same on working days and holidays.
- **Alternate Hypothesis (Ha):** The count of rented electric cycles is higher on working days than on holidays.

1. **Statistical Test Used:**

- **Two Sample Independent T-Test:** Given that the data consists of two categorical groups (working days and holidays) and the count of rented electric cycles (numerical), this test was applied. The assumption of independence between the groups was upheld.

1. **Significance Level:**

- A significance level of 0.05 (95% confidence interval) was chosen.

1. **Result Interpretation:**

- The computed p-value from the t-test was found to be less than 0.05.
- Therefore, we reject the null hypothesis. This indicates that working days have a statistically significant impact on the count of rented electric cycles. Specifically, the count of rented electric bikes is higher on working days compared to holidays.

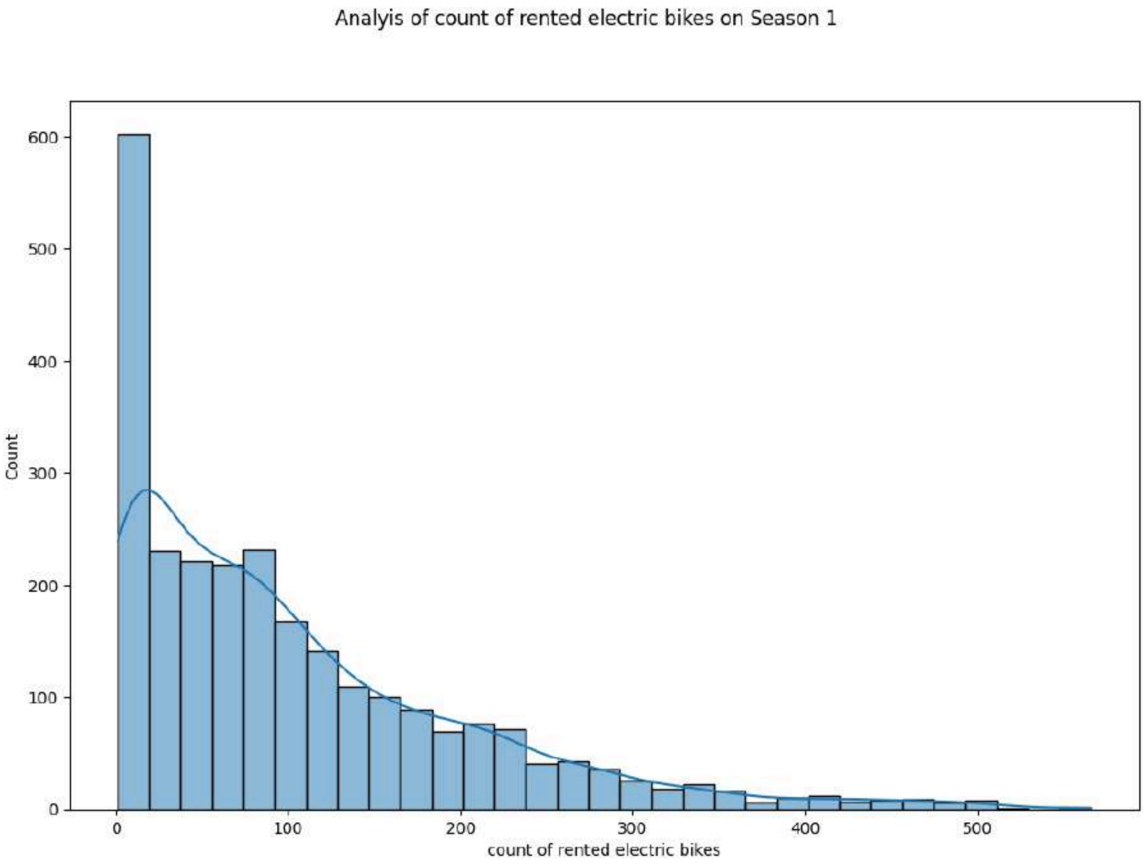
2. Whether Season has an effect on the count of rented electric bikes

In [49]:

```
# Creating two different dataframes for each season on count of renting bikes

season_1_df = final_df.loc[(final_df["season"]==1),"count"]
season_2_df = final_df.loc[(final_df["season"]==2),"count"]
season_3_df = final_df.loc[(final_df["season"]==3),"count"]
season_4_df = final_df.loc[(final_df["season"]==4),"count"]
```

In [50]:

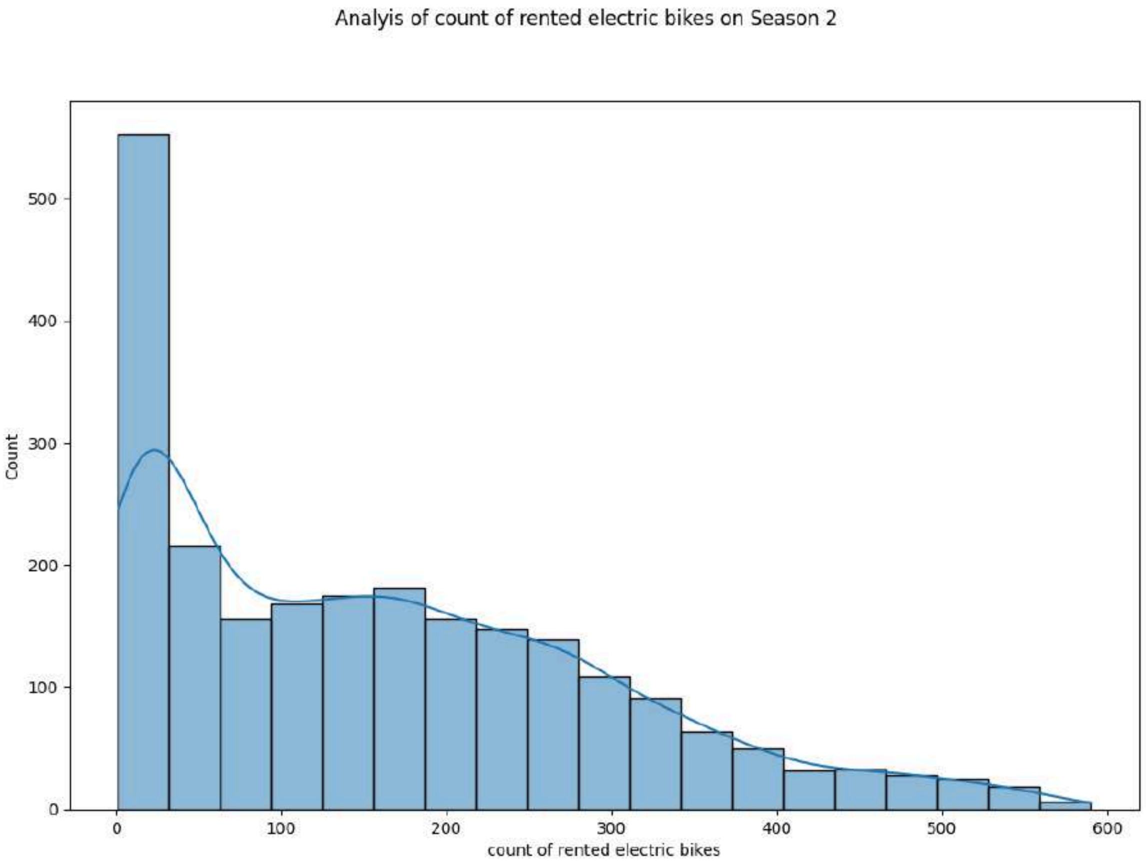
```
# checking for the dataset's normal distribution with Wilkin-Shapiro Test

# Null Hypothesis(H0) : Data is Normally distributed
# Alternate Hypothesis(Ha) : Data is not Normally distributed

plt.figure(figsize = (12,8))
sns.histplot(season_1_df,kde=True)
plt.suptitle("Analyis of count of rented electric bikes on Season 1")
plt.xlabel("count of rented electric bikes")
plt.show()

season_1_df = season_1_df.sample(100)
shapiro_stat,p_value = shapiro(season_1_df)
alpha = 0.05
print("Confidence Interval: 95%")
print("Wilkin-Shapiro test with Test Statistic: {}, and p-value: {}".format
(shapiro_stat,p_value))
if p_value < alpha:
    print("Reject Ho: Data is not Normally distributed")
else:
    print("Failed to reject Ho: Data is Normally distributed")
```



```
Confidence Interval: 95%
Wilkin-Shapiro test with Test Statistic: 0.8714165687561035, and p-value:
7.999452833473697e-08
Reject Ho: Data is not Normally distributed
```

In [51]:

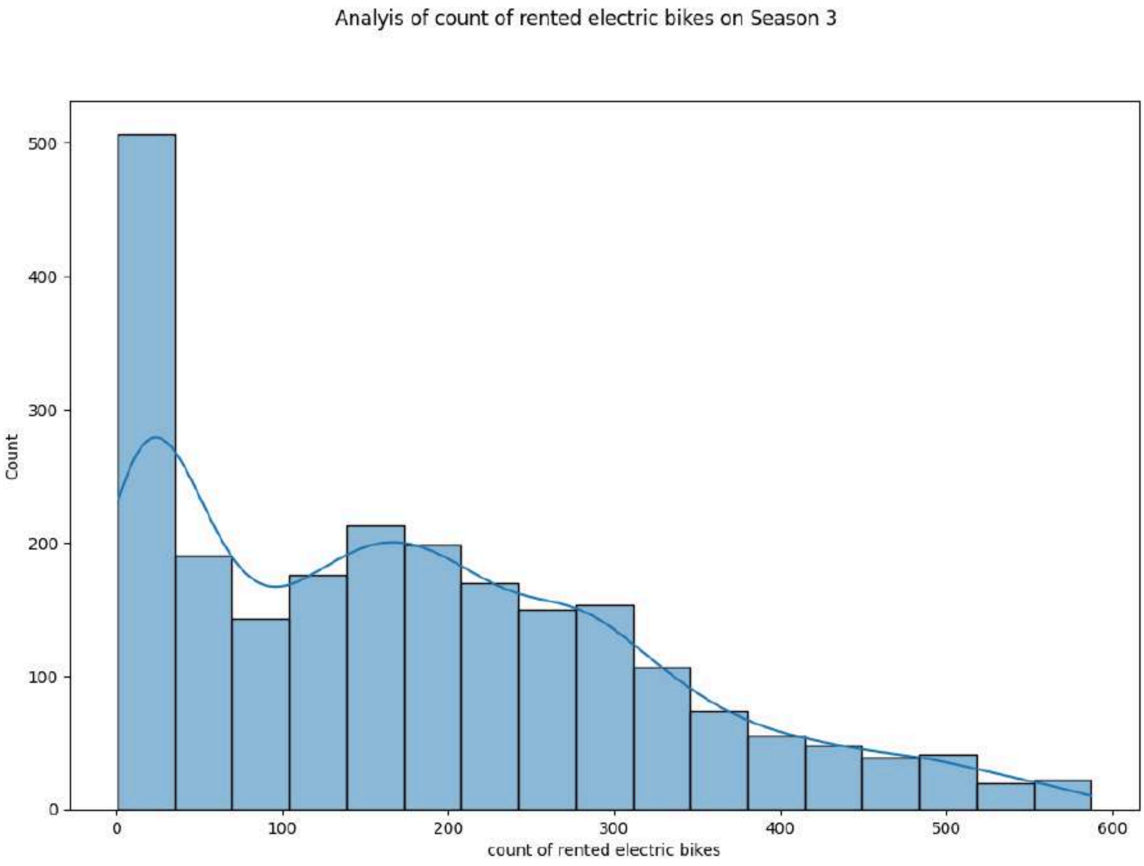
```
# checking for the dataset's normal distribution with Wilkin-Shapiro Test

# Null Hypothesis(H0) : Data is Normally distributed
# Alternate Hypothesis(Ha) : Data is not Normally distributed

plt.figure(figsize = (12,8))
sns.histplot(season_2_df,kde=True)
plt.suptitle("Analyis of count of rented electric bikes on Season 2")
plt.xlabel("count of rented electric bikes")
plt.show()

season_2_df = season_2_df.sample(100)
shapiro_stat,p_value = shapiro(season_2_df)
alpha = 0.05
print("Confidence Interval: 95%")
print("Wilkin-Shapiro test with Test Statistic: {}, and p-value: {}".format
(shapiro_stat,p_value))
if p_value < alpha:
    print("Reject Ho: Data is not Normally distributed")
else:
    print("Failed to reject Ho: Data is Normally distributed")
```



```
Confidence Interval: 95%
Wilkin-Shapiro test with Test Statistic: 0.9158322215080261, and p-value:
8.516610250808299e-06
Reject Ho: Data is not Normally distributed
```

In [52]:

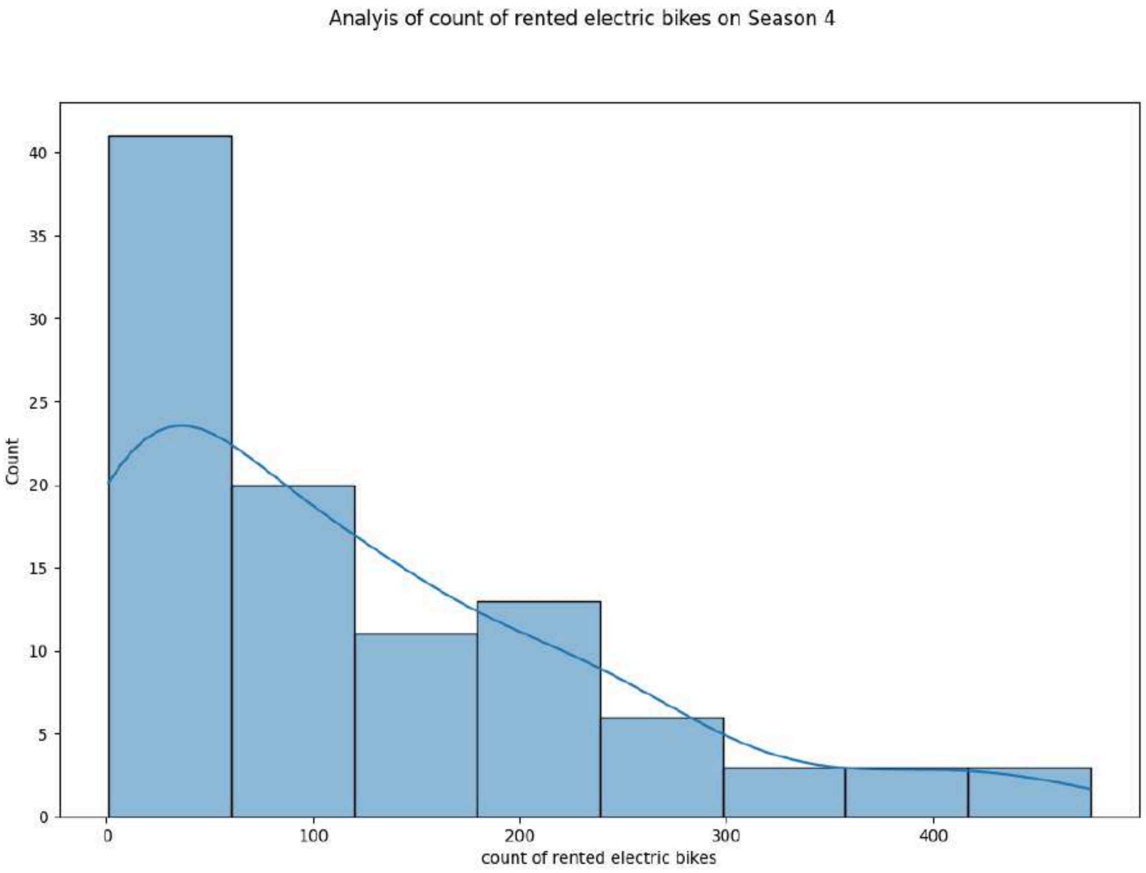
```
# checking for the dataset's normal distribution with Wilkin-Shapiro Test

# Null Hypothesis(H0) : Data is Normally distributed
# Alternate Hypothesis(Ha) : Data is not Normally distributed

plt.figure(figsize = (12,8))
sns.histplot(season_3_df,kde=True)
plt.suptitle("Analyis of count of rented electric bikes on Season 3")
plt.xlabel("count of rented electric bikes")
plt.show()

season_3_df = season_3_df.sample(100)
shapiro_stat,p_value = shapiro(season_3_df)
alpha = 0.05
print("Confidence Interval: 95%")
print("Wilkin-Shapiro test with Test Statistic: {}, and p-value: {}".format
(shapiro_stat,p_value))
if p_value < alpha:
    print("Reject Ho: Data is not Normally distributed")
else:
    print("Failed to reject Ho: Data is Normally distributed")
```



```
Confidence Interval: 95%
Wilkin-Shapiro test with Test Statistic: 0.934975266456604, and p-value: 9.
643857629271224e-05
Reject Ho: Data is not Normally distributed
```

In [53]:

```
# checking for the dataset's normal distribution with Wilkin-Shapiro Test

# Null Hypothesis(H0) : Data is Normally distributed
# Alternate Hypothesis(Ha) : Data is not Normally distributed

plt.figure(figsize = (12,8))
sns.histplot(season_1_df,kde=True)
plt.suptitle("Analyis of count of rented electric bikes on Season 4")
plt.xlabel("count of rented electric bikes")
plt.show()

season_4_df = season_4_df.sample(100)
shapiro_stat,p_value = shapiro(season_4_df)
alpha = 0.05
print("Confidence Interval: 95%")
print("Wilkin-Shapiro test with Test Statistic: {}, and p-value: {}".format
(shapiro_stat,p_value))
if p_value < alpha:
    print("Reject Ho: Data is not Normally distributed")
else:
    print("Failed to reject Ho: Data is Normally distributed")
```



```
Confidence Interval: 95%
Wilkin-Shapiro test with Test Statistic: 0.9245712757110596, and p-value:
2.4808976377244107e-05
Reject Ho: Data is not Normally distributed
```

```
In [54]: # Checking for equal variance among different groups with Levene's Test

         # Null Hypothesis(H0) : Variance among the groups are equal.
         # Alternate Hypothesis(Ha) : Variance among the groups are not equal.

         levene_stat,p_value = levene(season_1_df,season_2_df,season_3_df,season_4_d
         f)
         alpha = 0.05
         print("Confidence Interval: 95%")
         print("Levene test with Test Statistic: {}, and p-value: {}".format(levene_
         stat,p_value))
         if p_value < alpha:
             print("Reject Ho: Variance among the groups are not equal")
         else:
             print("Failed to reject Ho: Variance among the groups are equal")
```

```
Confidence Interval: 95%
Levene test with Test Statistic: 1.9838611088577554, and p-value: 0.1158325
8951615096
Failed to reject Ho: Variance among the groups are equal
```

Setting up Null and Alternate Hypothesis

Null Hypothesis (H0): The season does not have an impact on the count of rented electric bikes, i.e., the number of bikes rented is the same across different seasons.

Alternate Hypothesis (Ha): The season does have an impact on the count of rented electric bikes, i.e., the number of bikes rented differs across different seasons.

Significance Level (α): 0.05

Justification for Kruskal-Wallis Test:

- The data is not normally distributed.
- The assumption of equal variances among groups (seasons) is violated.
- Season is a categorical variable with more than two categories, and count of rented electric bikes is a numerical variable.
- Kruskal-Wallis Test: This non-parametric test is appropriate for comparing more than two independent groups (seasons) when the assumptions of ANOVA (parametric test) are not met.

Assumptions and Considerations:

- Data Distribution: The count of rented electric bikes may not follow a normal distribution.
- Equal Variances: Variance among the groups (seasons) is not assumed to be equal.
- Independence: Data points (counts of rented bikes) are independent within each season category.

Interpretation:

- If the p-value from the Kruskal-Wallis test is less than 0.05, we reject the null hypothesis.
- If the p-value is greater than or equal to 0.05, we fail to reject the null hypothesis.

In [55]:

```
#  Implementation of Kruskal-Wallis Test

test_statistic,p_value = kruskal(season_1_df,season_2_df,season_3_df,season_4_df)
print("Kruskal-Wallis Test with Test Statistic: {}, and p-value: {}".format(test_statistic,p_value))
alpha = 0.05
print("Confidence Interval: 95%")
if p_value < alpha:
    print("Reject Ho: Season does have impact on the count of rented electric bikes,i.e., No. of bikes are different in different seasons.")
else:
    print("Failed to reject Ho: Season doesn't have impact on the count of rented electric bikes,i.e., No. of bikes are same in different seasons")
```

```
Kruskal-Wallis Test with Test Statistic: 14.370877188376763, and p-value:
0.0024414240323126723
Confidence Interval: 95%
Reject Ho: Season does have impact on the count of rented electric bikes,i.
e., No. of bikes are different in different seasons.
```

In [56]:

```
#  Implementation of ANOVA (f_oneway)

test_statistic,p_value = f_oneway(season_1_df,season_2_df,season_3_df,season_4_df)
print("ANOVA with Test Statistic: {}, and p-value: {}".format(test_statistic,p_value))
alpha = 0.05
print("Confidence Interval: 95%")
if p_value < alpha:
    print("Reject Ho: Season does have impact on the count of rented electric bikes,i.e., No. of bikes are different in different seasons.")
else:
    print("Failed to reject Ho: Season doesn't have impact on the count of rented electric bikes,i.e., No. of bikes are same in different seasons")
```

```
ANOVA with Test Statistic: 4.580626110470527, and p-value: 0.00362709558280
48247
Confidence Interval: 95%
Reject Ho: Season does have impact on the count of rented electric bikes,i.
e., No. of bikes are different in different seasons.
```

Analysis and Explanation

1. **Dataset Preparation:**

- Four separate datasets were created from final_df, each corresponding to one of the four seasons (spring, summer, fall, winter).

1. **Normality and Equal Variance Checks:**

- Shapiro-Wilk Test: Normality of the data distribution was assessed using the Shapiro-Wilk test. Results indicated that the data did not follow a normal distribution for each season.
- Levene's Test: Equal variance among the groups (seasons) was checked using Levene's test. The test showed that variances among the groups were not equal.

1. **Null and Alternative Hypotheses:**

- Null Hypothesis (H0): The season does not have an impact on the count of rented electric bikes, i.e., the number of bikes rented is the same across different seasons.
- Alternate Hypothesis (Ha): The season does have an impact on the count of rented electric bikes, i.e., the number of bikes rented differs across different seasons.

1. **Statistical Tests Used:**

- Kruskal-Wallis Test: Given the violations of normality and equal variance assumptions for ANOVA, the Kruskal-Wallis test was chosen. This non-parametric test compares the distributions of numerical data across different groups (seasons).
- ANOVA (f_oneway): Despite the violations, ANOVA was also conducted for comparison. However, its results are considered cautiously due to the violated assumptions.

1. Significance Level:

- A significance level of 0.05 (95% confidence interval) was used for both tests.

1. Result Interpretation:

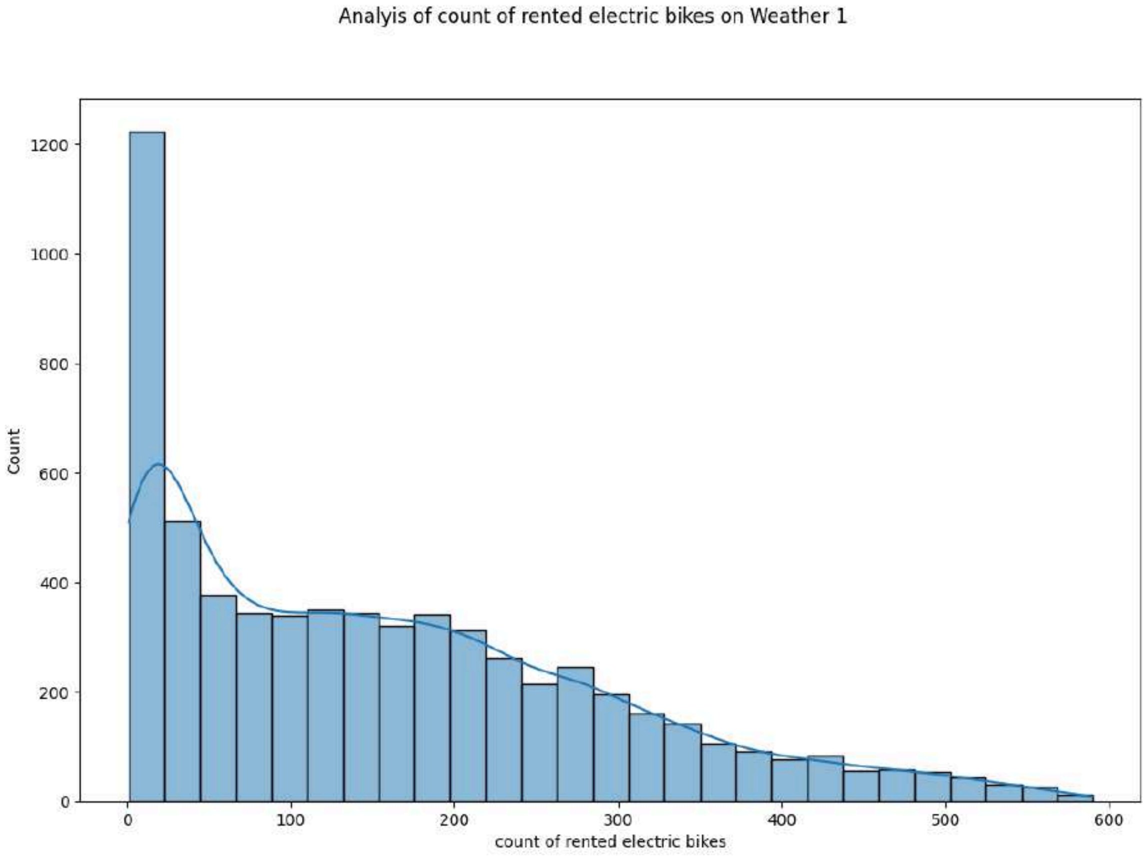
- The computed p-values from both the Kruskal-Wallis test and ANOVA were found to be less than 0.05.
- Therefore, we reject the null hypothesis. This indicates that season has a statistically significant impact on the count of rented electric bikes. Specifically, the number of bikes rented varies across different seasons.

3. Whether Weather has an effect on the count of rented electric bikes

In [57]:

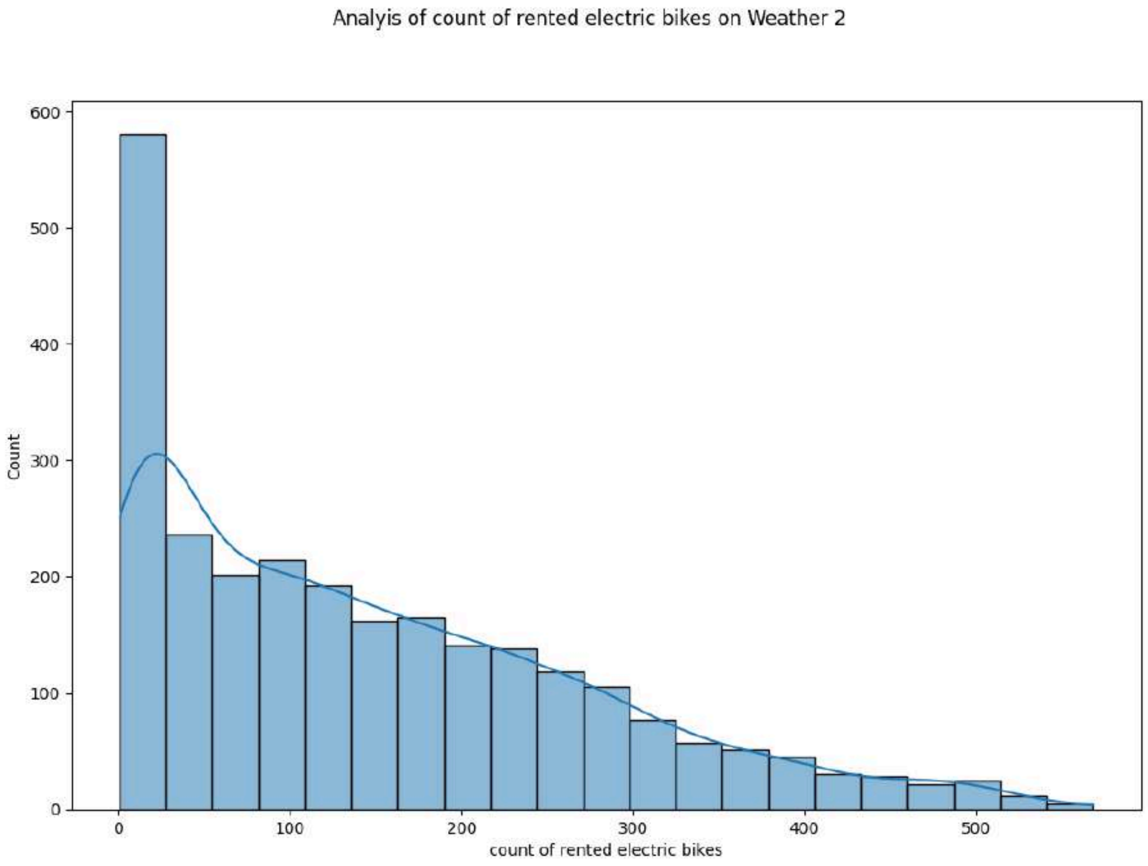
```
# Creating two different dataframes for each weather on count of renting bikes

weather_1_df = final_df.loc[(final_df["weather"]==1),"count"]
weather_2_df = final_df.loc[(final_df["weather"]==2),"count"]
weather_3_df = final_df.loc[(final_df["weather"]==3),"count"]
weather_4_df = final_df.loc[(final_df["weather"]==4),"count"]
```

In [58]:
```
# checking for the dataset's normal distribution with Wilkin-Shapiro Test

# Null Hypothesis(H0) : Data is Normally distributed
# Alternate Hypothesis(Ha) : Data is not Normally distributed

plt.figure(figsize = (12,8))
sns.histplot(weather_1_df,kde=True)
plt.suptitle("Analyis of count of rented electric bikes on Weather 1")
plt.xlabel("count of rented electric bikes")
plt.show()

weather_1_sample_df = weather_1_df.sample(100)
shapiro_stat,p_value = shapiro(weather_1_sample_df)
alpha = 0.05
print("Confidence Interval: 95%")
print("Wilkin-Shapiro test with Test Statistic: {}, and p-value: {}".format
(shapiro_stat,p_value))
if p_value < alpha:
    print("Reject Ho: Data is not Normally distributed")
else:
    print("Failed to reject Ho: Data is Normally distributed")
```



```
Confidence Interval: 95%
Wilkin-Shapiro test with Test Statistic: 0.8950494527816772, and p-value:
8.332827405865828e-07
Reject Ho: Data is not Normally distributed
```

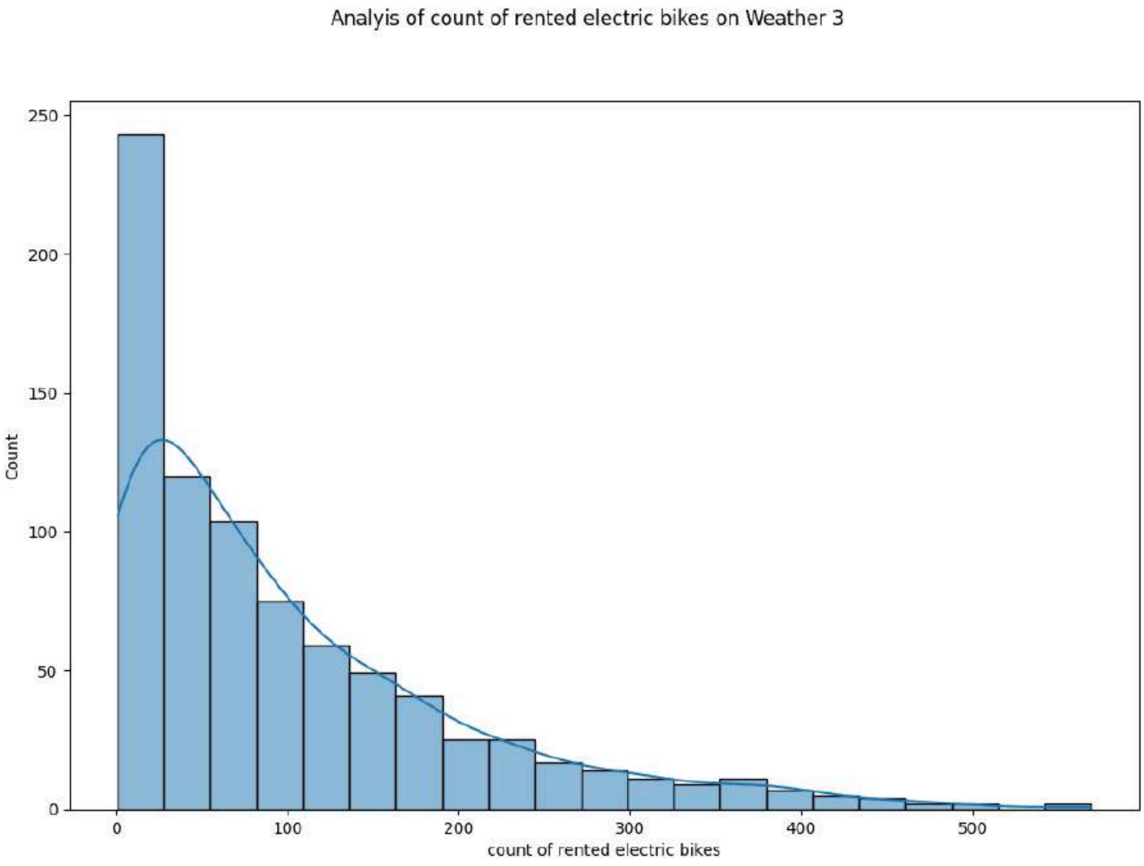
```
In [59]: # checking for the dataset's normal distribution with Wilkin-Shapiro Test

         # Null Hypothesis(H0) : Data is Normally distributed
         # Alternate Hypothesis(Ha) : Data is not Normally distributed

         plt.figure(figsize = (12,8))
         sns.histplot(weather_2_df,kde=True)
         plt.suptitle("Analyis of count of rented electric bikes on Weather 2")
         plt.xlabel("count of rented electric bikes")
         plt.show()

         weather_2_sample_df = weather_2_df.sample(100)
         shapiro_stat,p_value = shapiro(weather_2_sample_df)
         alpha = 0.05
         print("Confidence Interval: 95%")
         print("Wilkin-Shapiro test with Test Statistic: {}, and p-value: {}".format
         (shapiro_stat,p_value))
         if p_value < alpha:
             print("Reject Ho: Data is not Normally distributed")
         else:
             print("Failed to reject Ho: Data is Normally distributed")
```



```
Confidence Interval: 95%
Wilkin-Shapiro test with Test Statistic: 0.8970624208450317, and p-value:
1.0313206075807102e-06
Reject Ho: Data is not Normally distributed
```

In [60]:

```
# checking for the dataset's normal distribution with Wilkin-Shapiro Test

# Null Hypothesis(H0) : Data is Normally distributed
# Alternate Hypothesis(Ha) : Data is not Normally distributed

plt.figure(figsize = (12,8))
sns.histplot(weather_3_df,kde=True)
plt.suptitle("Analyis of count of rented electric bikes on Weather 3")
plt.xlabel("count of rented electric bikes")
plt.show()

weather_3_sample_df = weather_3_df.sample(100)
shapiro_stat,p_value = shapiro(weather_3_sample_df)
alpha = 0.05
print("Confidence Interval: 95%")
print("Wilkin-Shapiro test with Test Statistic: {}, and p-value: {}".format
(shapiro_stat,p_value))
if p_value < alpha:
    print("Reject Ho: Data is not Normally distributed")
else:
    print("Failed to reject Ho: Data is Normally distributed")
```



```
Confidence Interval: 95%
Wilkin-Shapiro test with Test Statistic: 0.8293532729148865, and p-value:
2.1985842035832093e-09
Reject Ho: Data is not Normally distributed
```

In [61]:

```
# Checking for equal variance among different groups with Levene's Test

# Null Hypothesis(H0) : Variance among the groups are equal.
# Alternate Hypothesis(Ha) : Variance among the groups are not equal.

levene_stat,p_value = levene(weather_1_df,weather_2_df,weather_3_df,weather
_4_df)
alpha = 0.05
print("Confidence Interval: 95%")
print("Levene test with Test Statistic: {}, and p-value: {}".format(levene_
stat,p_value))
if p_value < alpha:
    print("Reject Ho: Variance among the groups are not equal")
else:
    print("Failed to reject Ho: Variance among the groups are equal")
```

```
Confidence Interval: 95%
Levene test with Test Statistic: 44.654420886537366, and p-value: 1.1862780
886165858e-28
Reject Ho: Variance among the groups are not equal
```

Setting up Null and Alternate Hypothesis

Null Hypothesis (H0): The weather does not have an impact on the count of rented electric bikes, i.e., the number of bikes rented is the same across different weather conditions.

Alternate Hypothesis (Ha): The weather does have an impact on the count of rented electric bikes, i.e., the number of bikes rented differs across different weather conditions.

Significance Level (α): 0.05

Justification for Kruskal-Wallis Test:

- The data is not normally distributed.
- Variance among the groups (weather conditions) is not equal.
- Weather is a categorical variable with more than two categories (1, 2, 3), and count of rented electric bikes is a numerical variable.
- Kruskal-Wallis Test: This non-parametric test is appropriate for comparing more than two independent groups (weather conditions) when the assumptions of ANOVA (parametric test) are violated.

Assumptions and Considerations:

- Data Distribution: The count of rented electric bikes may not follow a normal distribution.
- Equal Variances: Variance among the groups (weather conditions) is not assumed to be equal.
- Independence: Data points (counts of rented bikes) are independent within each weather category.

Interpretation:

- If the p-value from the Kruskal-Wallis test is less than 0.05, we reject the null hypothesis.
- If the p-value is greater than or equal to 0.05, we fail to reject the null hypothesis.

In [62]:

```
#  Implementation of Kruskal-Wallis Test

test_statistic,p_value = kruskal(weather_1_df,weather_2_df,weather_3_df,weather_4_df)
print("Kruskal-Wallis Test with Test Statistic: {}, and p-value: {}".format(test_statistic,p_value))
alpha = 0.05
print("Confidence Interval: 95%")
if p_value < alpha:
    print("Reject Ho: Weather does have impact on the count of rented electric bikes,i.e., No. of bikes is different in different weather.")
else:
    print("Failed to reject Ho: Weather doesn't have impact on the count of rented electric bikes,i.e., No. of bikes is same in different weather")
```

```
Kruskal-Wallis Test with Test Statistic: 133.4609634790672, and p-value: 9.708574804186936e-29
Confidence Interval: 95%
Reject Ho: Weather does have impact on the count of rented electric bikes, i.e., No. of bikes is different in different weather.
```

In [63]:

```
#  Implementation of ANOVA(f_oneway)

test_statistic,p_value = f_oneway(weather_1_df,weather_2_df,weather_3_df,weather_4_df)
print("Kruskal-Wallis Test with Test Statistic: {}, and p-value: {}".format(test_statistic,p_value))
alpha = 0.05
print("Confidence Interval: 95%")
if p_value < alpha:
    print("Reject Ho: Weather does have impact on the count of rented electric bikes,i.e., No. of bikes is different in different weather.")
else:
    print("Failed to reject Ho: Weather doesn't have impact on the count of rented electric bikes,i.e., No. of bikes is same in different weather")
```

```
Kruskal-Wallis Test with Test Statistic: 46.45209330731525, and p-value: 8.465264561246368e-30
Confidence Interval: 95%
Reject Ho: Weather does have impact on the count of rented electric bikes, i.e., No. of bikes is different in different weather.
```

Analysis and Explanation

1. **Dataset Preparation:**

- Four separate datasets were created from final_df, each corresponding to one of the four weather categories (1, 2, 3, 4).

1. **Normality and Equal Variance Checks:**

- Shapiro-Wilk Test: Normality of the data distribution was assessed using the Shapiro-Wilk test. Results indicated that the data did not follow a normal distribution for each weather category.
- Levene's Test: Equal variance among the groups (weather categories) was checked using Levene's test. The test showed that variances among the groups were not equal.

1. **Null and Alternative Hypotheses:**

- **Null Hypothesis (H0):** The weather does not have an impact on the count of rented electric bikes, i.e., the number of bikes rented is the same across different weather conditions.
- **Alternate Hypothesis (Ha):** The weather does have an impact on the count of rented electric bikes, i.e., the number of bikes rented differs across different weather conditions.

1. **Statistical Tests Used:**

- Kruskal-Wallis Test: Given the violations of normality and equal variance assumptions for ANOVA, the Kruskal-Wallis test was chosen. This non-parametric test compares the distributions of numerical data across different groups (weather conditions).
- ANOVA (f_oneway): Despite the violations, ANOVA was also conducted for comparison. However, its results are considered cautiously due to the violated assumptions.

1. **Significance Level:**

- A significance level of 0.05 (95% confidence interval) was used for both tests.

1. **Result Interpretation:**

- The computed p-values from both the Kruskal-Wallis test and ANOVA were found to be less than 0.05.
- Therefore, we reject the null hypothesis. This indicates that weather has a statistically significant impact on the count of rented electric bikes. Specifically, the number of bikes rented varies across different weather conditions.

4. Whether weather is dependent on Season

In [64]:

```
# Creating a separate dataframe consisting of season,weather and corresponding count, and then calculating using crosstab their corresponding mean and putting them into a 2D numpy array.
weather_df = final_df[["season","weather","count"]]
weather_df = pd.crosstab(index=weather_df['season'],columns=weather_df['weather'],values = weather_df['count'], aggfunc = 'mean').fillna(0).reset_index(drop=True)
data=weather_df.values
```

Setting up Null and Alternate Hypothesis

Null Hypothesis (H0): Weather conditions are independent of the season.

Alternate Hypothesis (Ha): Weather conditions are dependent on the season.

Significance Level (α): 0.05

Justification for Chi-square Test:

- We have categorical data (weather: 1, 2, 3, 4) and another categorical variable (season: spring, summer, fall, winter).
- Chi-square test of independence (chi2_contingency) is appropriate to determine whether there is a relationship between two categorical variables.
- The test assesses whether the distribution of weather conditions varies significantly across different seasons.

Assumptions and Considerations:

- Independence: Data points (weather conditions) are independent within each season.
- Expected Frequencies: Each cell in the contingency table should have an expected frequency of at least 5 for the Chi-square test to be valid.

Interpretation:

- If the p-value from the Chi-square test is less than 0.05, we reject the null hypothesis.
- If the p-value is greater than or equal to 0.05, we fail to reject the null hypothesis.

In [65]:

```
# Implementation of Chi-Square Test of Independence

test_statistic,p_value,dof,exp_freq = chi2_contingency(data)
print("Chi-Square Test of Independence with Test Statistic: {}, p-value:
{}, and Degree of Freedom: {}".format(test_statistic,p_value,dof))
alpha = 0.05
print("Confidence Interval: 95%")
if p_value < alpha:
    print("Reject Ho: Weather does depend on the season.")
else:
    print("Failed to reject Ho: Weather doesn't depend on the season.")
```

```
Chi-Square Test of Independence with Test Statistic: 574.0436779380453, p-v
alue: 7.769030862492005e-118, and Degree of Freedom: 9
Confidence Interval: 95%
Reject Ho: Weather does depend on the season.
```

Analysis and Explanation

1. Dataset Preparation:

A dataset was created from final_df containing the variables season and weather, along with their corresponding count of rented bikes.

2. Crosstabulation:

- Cross-tabulation (crosstab): A crosstab was performed on the dataset, creating a contingency table with seasons as rows and weather conditions as columns. This table was then converted into a 2D numpy array for further analysis.

3. Null and Alternative Hypotheses:

- Null Hypothesis (H0): Weather conditions are independent of the season.
- Alternate Hypothesis (Ha): Weather conditions are dependent on the season.

4. Statistical Test Used:

- Chi-Square Test of Independence: Given the categorical nature of both variables (season and weather) and the need to assess their relationship, the Chi-square test of independence was applied. This test evaluates whether there is a significant association between two categorical variables.

5. Significance Level:

- A significance level of 0.05 (95% confidence interval) was chosen to determine the threshold for statistical significance.

6. Result Interpretation:

- The computed p-value from the Chi-square test was found to be less than 0.05.
- Therefore, we reject the null hypothesis. This indicates that there is a statistically significant dependence between weather conditions and seasons. In other words, the distribution of weather conditions varies significantly across different seasons.

Business Insights

1. Weather and Seasonal Demand:

- Customers prefer renting electric bikes during clear or cloudy weather, primarily from January to August. This seasonal trend suggests focusing bike availability and services during these periods to meet high demand.

2. Impact of Adverse Weather Conditions:

- Rental numbers decrease significantly during rain, thunderstorms, snow, or foggy conditions. Additionally, when humidity is below 20%, bike rentals are notably lower. Understanding these patterns can help optimize bike fleet management based on weather forecasts.

3. Temperature and Windspeed

- Bike rentals decrease when temperatures drop below 10°C. Similarly, high windspeeds exceeding 35 units also correlate with lower rental numbers. Monitoring these conditions can aid in adjusting service offerings during adverse weather.

4. Differential Renting Behavior:

- Registered users prefer renting bikes on working non-holidays, particularly during office start and end hours. In contrast, casual riders rent more bikes on holidays or non-working days. Tailoring services to these distinct consumer behaviors can enhance customer satisfaction and retention strategies.

5. Customer Segmentation and Service Planning:

- There is a significant disparity between registered riders and casual riders, with registered users constituting a larger portion of rentals. This insight underscores the importance of customer retention strategies and optimizing services based on user preferences.

6. Weather Dependency on Season:

- Hypothesis testing confirms that weather conditions depend on the season. This understanding is crucial for anticipating and adapting to seasonal variations in bike rental demand.

Recommendations

1. Seasonal Bike Allocation:

- Summer and Fall Seasons: Increase bike inventory during these seasons, particularly during clear or cloudy weather conditions. This aligns with higher rental demand observed during these periods.

2. Optimized Fleet Management:

- Working Days: Allocate a higher number of bikes on weekdays to cater to registered users, especially during office start and end hours. Adjust bike availability to match peak commuting times.
- Holidays: Maintain a nominal fleet size to cater to casual riders who are more active on holidays or non-working days.

3. Weather-Based Maintenance Strategy:

- Low Temperature Days (Below 10°C): Consider temporary removal of bikes for maintenance during days with extremely low temperatures to prevent operational issues and ensure bike readiness.
- High Windspeed or Adverse Weather: Implement proactive maintenance schedules during days with windspeed exceeding 35 units or during thunderstorms. This precautionary measure ensures bike safety and operational efficiency during challenging weather conditions.

Strategic Implementation

Implementing these recommendations can enhance operational efficiency, improve customer satisfaction, and optimize resource allocation based on seasonal and weather-related demand patterns. By focusing on proactive maintenance and strategic fleet management, the company can maximize rental availability during peak demand periods while ensuring operational reliability.