DSC630 Week3 - Assignment

Guruprasad Velikadu Krishnamoorthy

In [1]:
```
# Importing the Required libraries
import pandas as pd
import numpy as np
import os
import sys
import re
from datetime import datetime
# Importing the required packages for plotting graphs
import matplotlib.pyplot as plt
import seaborn as sns
import cufflinks as cf
import chart_studio.plotly as py
import plotly.express as px
import plotly.graph_objects as go
import plotly.io as pio
#pio.renderers.default = "notebook+pdf"
pio.renderers.default = "plotly_mimetype+notebook"
# Importing the required packages for Model building
import statsmodels.formula.api as smf
import patsy
```

In [2]:
```
# Setting global options for the notebook such as maxrows
pd.set_option('display.max_columns', 50)
pd.set_option('display.max_colwidth', None)
pd.set_option("display.max_rows", 100)
import warnings
warnings.filterwarnings('ignore')
```

1. Creating Dataframe from source data file

In [3]:
```
# Importing the Dataset
path=os.getcwd()
# Assigning a path for the file
dodgers_file_path=path+"\\dodgers-2022.csv"
```

In [4]:
```
# Loading the source file into Pandas DataFrame
dodgers_df_orig=pd.read_csv(dodgers_file_path)
# Printing the shape of the dataframe
dodgers_df_orig.shape
```

Out[4]: (81, 12)

In [5]:
```
# Making a copy of the original Dataframe
dodgers_df=dodgers_df_orig.copy()
# Printing top 5 rows of the Dataframe
dodgers_df.head()
```

Out[5]:

	month	day	attend	day_of_week	opponent	temp	skies	day_night	cap	shirt	fireworks	bobblehead
0	APR	10	56000	Tuesday	Pirates	67	Clear	Day	NO	NO	NO	NO
1	APR	11	29729	Wednesday	Pirates	58	Cloudy	Night	NO	NO	NO	NO
2	APR	12	28328	Thursday	Pirates	57	Cloudy	Night	NO	NO	NO	NO
3	APR	13	31601	Friday	Padres	54	Cloudy	Night	NO	NO	YES	NO
4	APR	14	46549	Saturday	Padres	57	Cloudy	Night	NO	NO	NO	NO

Goal of this Project: To make a recommendation to management on how to improve attendance

Overall Plan to achieve the Goal:

1. To perform EDA and build various visualizations in python to identify hidden patterns in the data.

2. Based on the outcome of the visualizations, build a model to identify the features that can improve the overall attendance.

2. Exploratory Data Analysis

In [6]:
```
# Getting a statistical summary of the dataframe
dodgers_df.describe()
```

Out[6]:

	day	attend	temp
count	81.000000	81.000000	81.000000
mean	16.135802	41040.074074	73.148148
std	9.605666	8297.539460	8.317318
min	1.000000	24312.000000	54.000000
25%	8.000000	34493.000000	67.000000
50%	15.000000	40284.000000	73.000000
75%	25.000000	46588.000000	79.000000
max	31.000000	56000.000000	95.000000

In [7]:
```
# Getting a statistical info of the dataframe like column types and non-null values
dodgers_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 81 entries, 0 to 80
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   month        81 non-null     object
 1   day          81 non-null     int64 
 2   attend       81 non-null     int64 
 3   day_of_week  81 non-null     object
 4   opponent     81 non-null     object
 5   temp         81 non-null     int64 
 6   skies        81 non-null     object
 7   day_night    81 non-null     object
 8   cap          81 non-null     object
 9   shirt        81 non-null     object
 10  fireworks    81 non-null     object
 11  bobblehead   81 non-null     object
dtypes: int64(3), object(9)
memory usage: 7.7+ KB
```

In [8]:
```
# Checking for nulls in each column
dodgers_df.isna().sum()
```

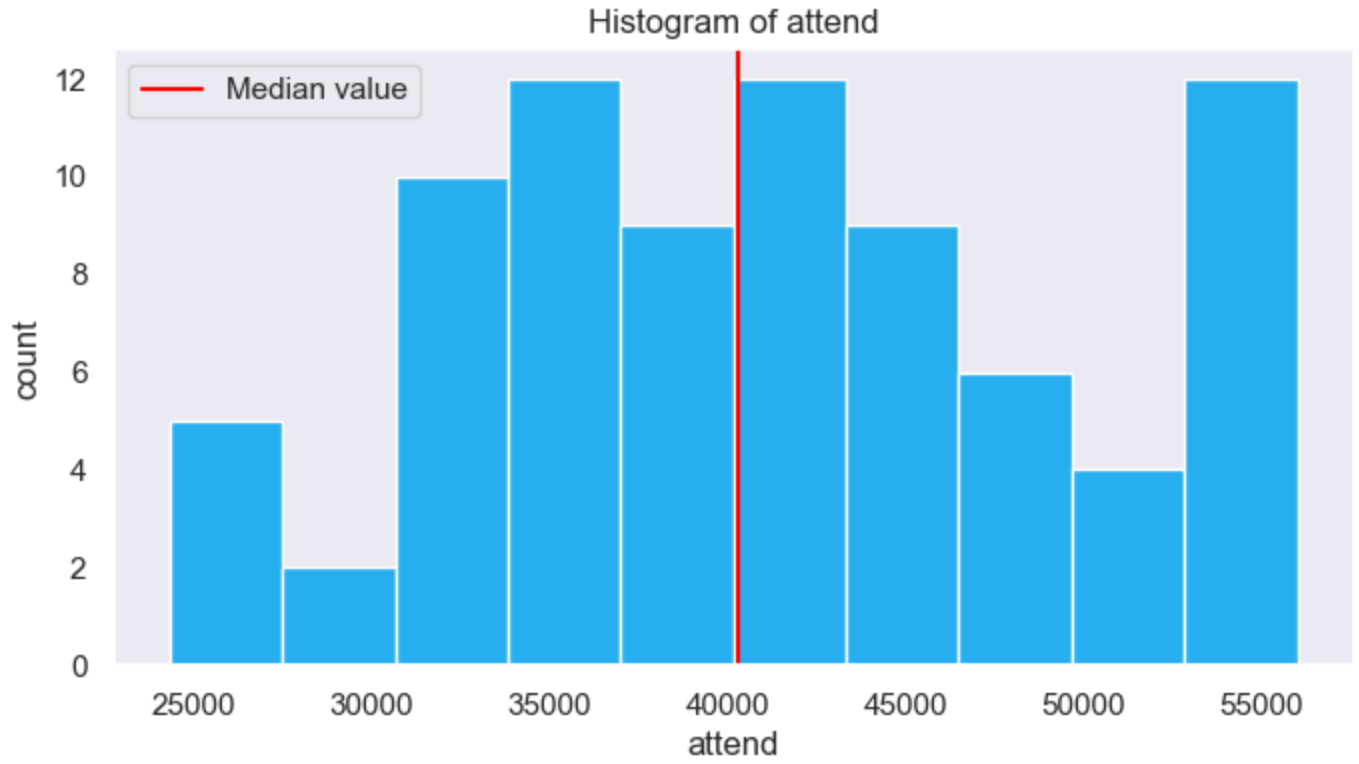
Out[8]:
```
month          0
day            0
attend         0
day_of_week    0
opponent       0
temp           0
skies          0
day_night      0
cap            0
shirt          0
fireworks      0
bobblehead     0
dtype: int64
```

2.1. Histogram of attendance and temperature

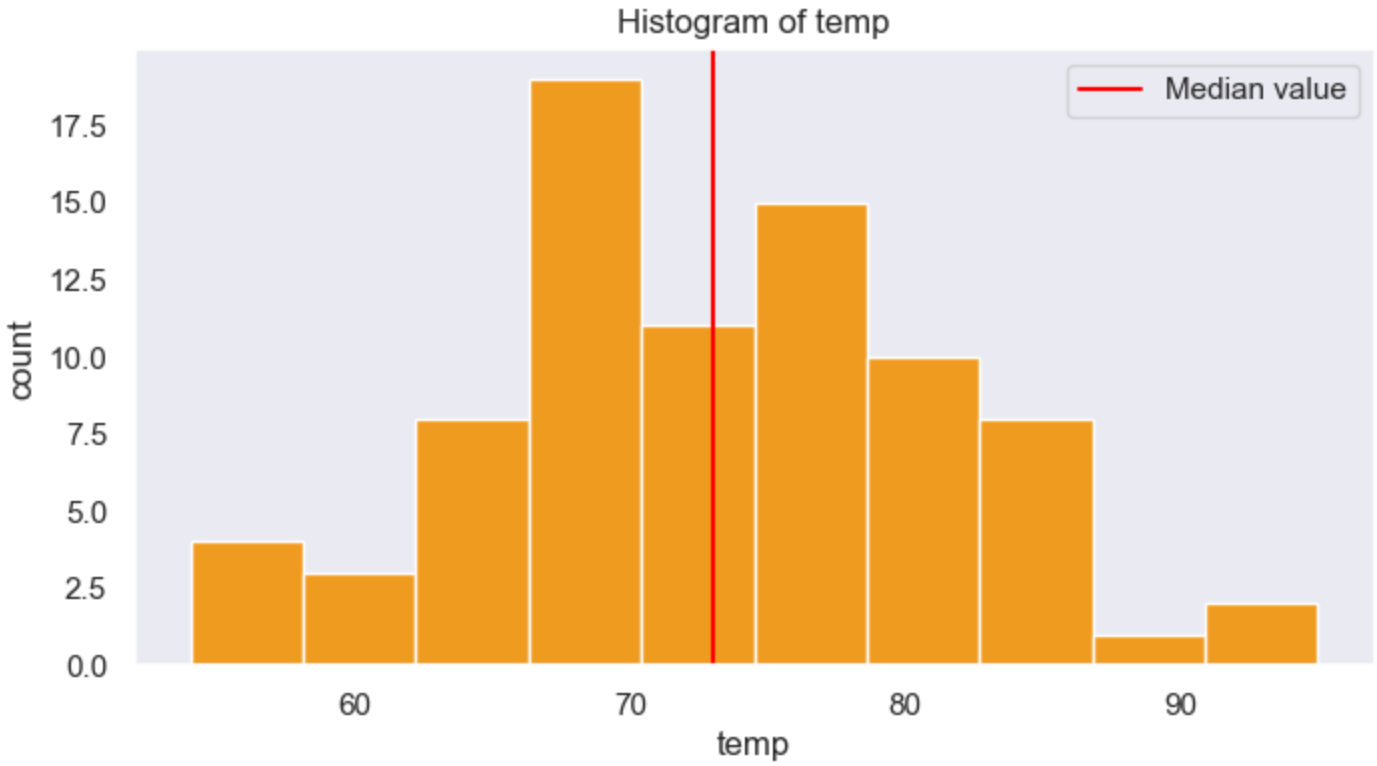
In [9]:
```
# Getting the list of Numeric columns that should be represented in the plot
selected_cols=["attend","temp"]
```

```
sns.set(font_scale=1)
# Setting the colors to be used in the plot
colors=[ "#27aeef", "#ef9b20"]
# Enumerating through the columns and creating Histogram for each column
for inx,col in enumerate(selected_cols):
    # Computing the median value
    median_val = dodgers_df[col].median()
    plt.figure(figsize = (8, 4))
    color = 'Red'
    # Plotting the Median line
    print(f"The Median of {col} is {median_val}")
    plt.axvline(median_val, color=color,label='Median value')
    dodgers_df[col].hist(grid=False,color= colors[inx])
    # Plotting the title and the labels
    plt.xlabel(dodgers_df[col].name)
    plt.ylabel('count')
    plt.legend()
    plt.title('Histogram of '+ col)
    plt.show()
```

```
The Median of attend is 40284.0
```

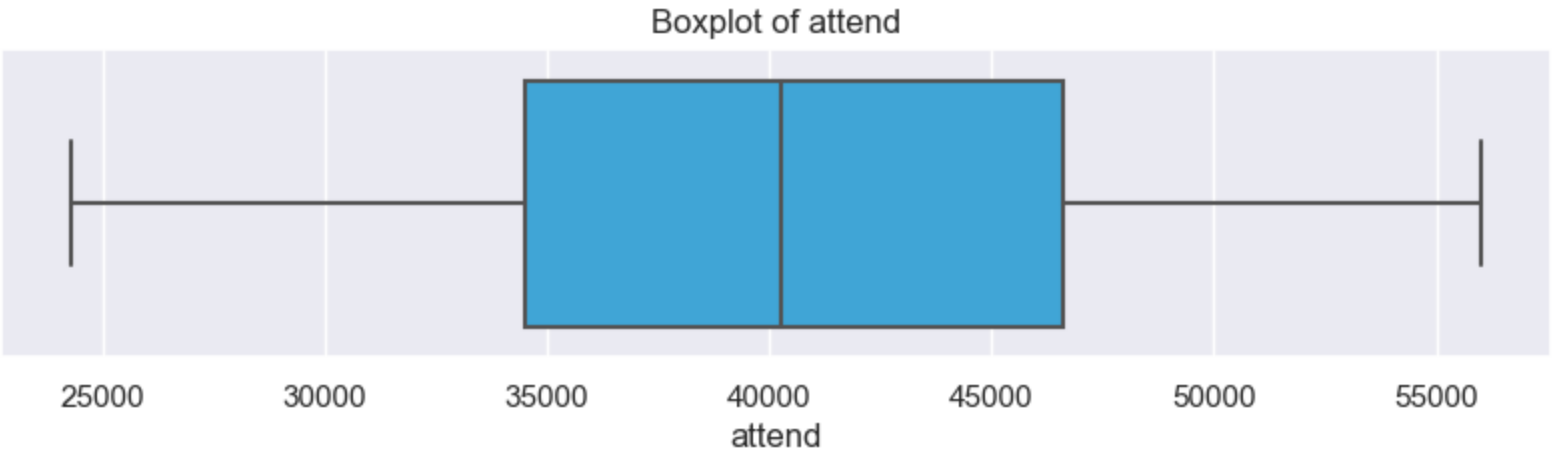


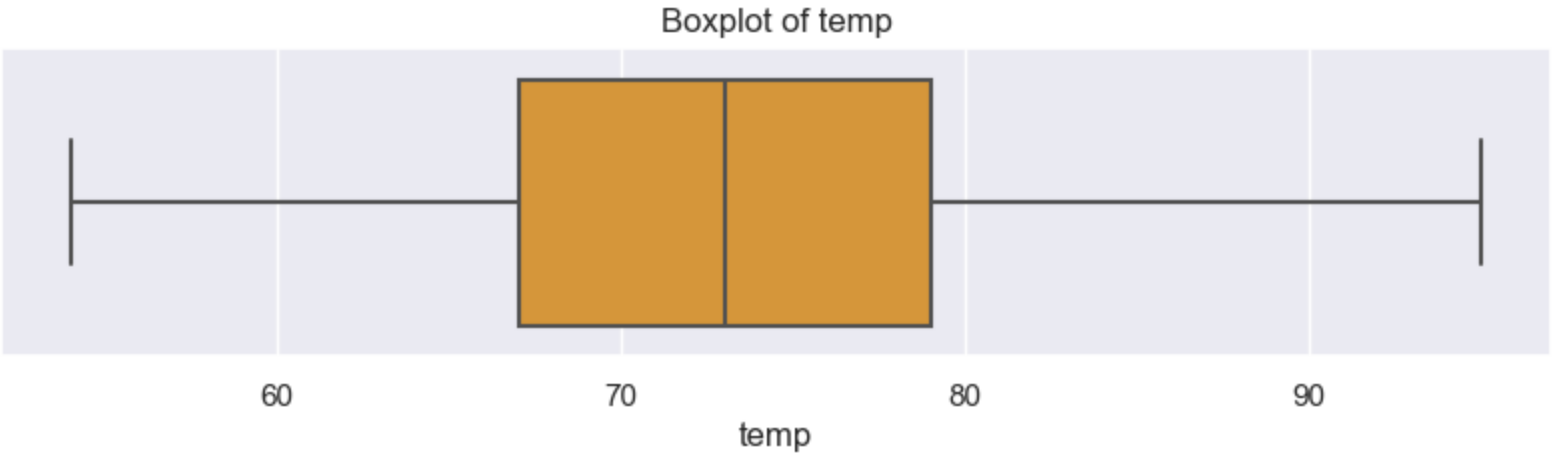
The Median of temp is 73.0



2.2. Boxplot of attendance and temperature

In [10]:
```
# Looping through each column in selected columns list and creating Box plot for each
for inx,col in enumerate(selected_cols):
    # Setting the Title of the plot
    plt.figure(figsize = (10,2))
    plt.title('Boxplot of '+ col)
    # creating boxplot
    sns.boxplot(x=dodgers_df[col],color= colors[inx])
    plt.show()
```



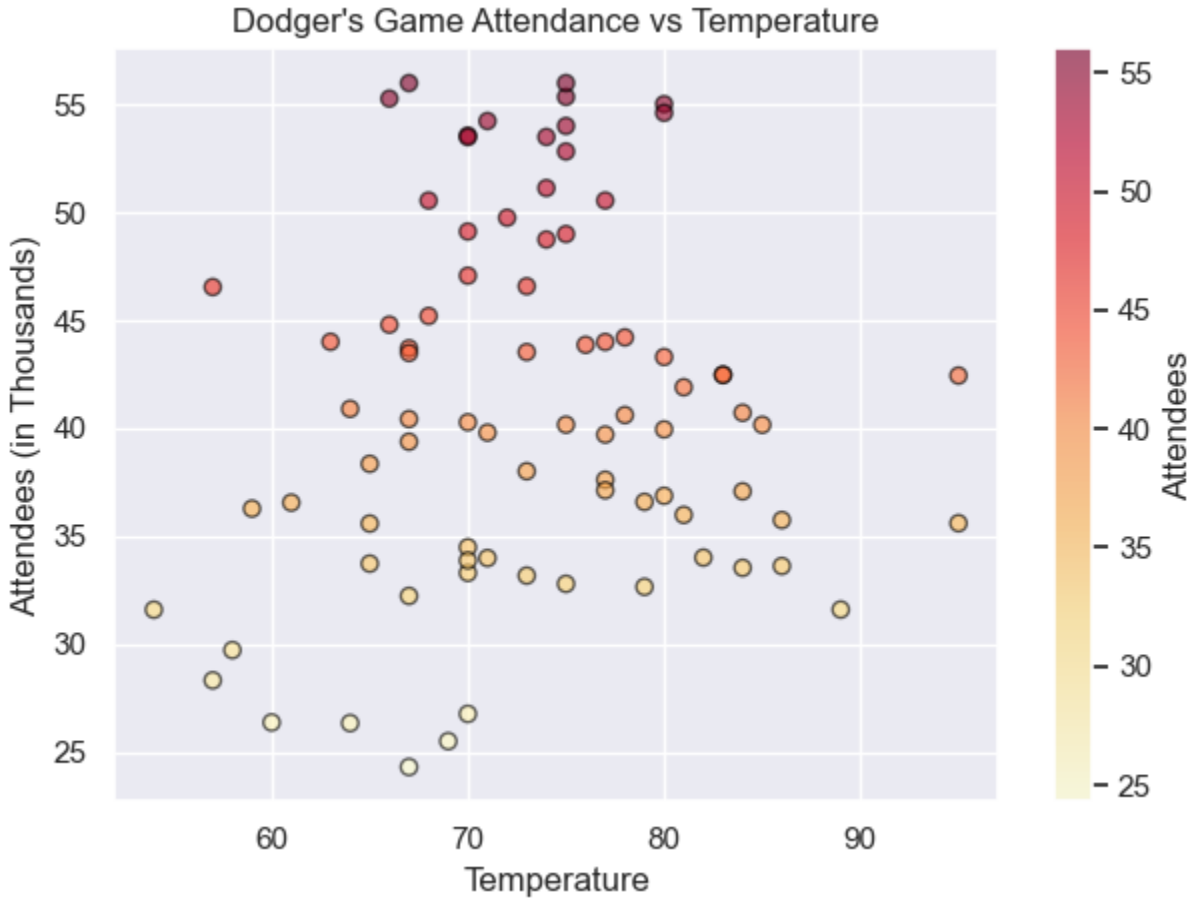


Observations from Histogram and Boxplot: The plot suggests that the median value of attendance is 40284 and the temperature is 73 degrees farenheit. Though the distribution does not appear to be a normal distribution, there are no outliers.

2.3. Scatter Plot of attendance vs temperature

In [11]:
```
# Creating  a scatter plot of Temperature versus attendance.
plt.scatter(dodgers_df["temp"],dodgers_df["attend"]/1000,s=35,alpha=0.6,cmap='YlOrRd',c=dodgers_df["attend"]
/1000,edgecolor="black")
# Setting color bar
cbar=plt.colorbar()
# Adding ColorBar label
```

```
cbar.set_label("Attendees")
plt.title("Dodger's Game Attendance vs Temperature")
plt.xlabel('Temperature')
plt.ylabel('Attendees (in Thousands)')
plt.tight_layout()
plt.show()
```



Observations from Scatter Plot: The scatter plot indicates that the attendance improved as the temperature increased suggesting a positive relation between the two. Also, most people attended the games when the temperature was between 65 to 80 degrees. ***This indicates that Temperature may be an important factor for the attendance.***

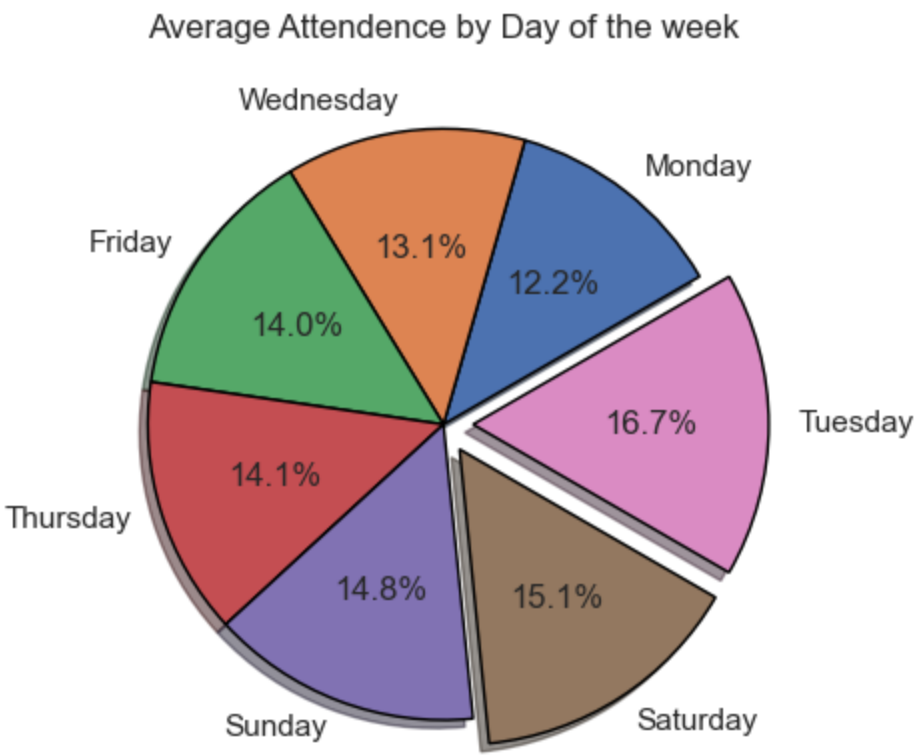
2.4. Pieplot of Game attendance by day of the week

In [12]:
```
# Creating a new dataframe by grouping the dat by the day of week and computing the mean
dodgers_group_df1=dodgers_df.groupby(['day_of_week']).mean()['attend'].sort_values().reset_index()
dodgers_group_df1
```

Out[12]:

	day_of_week	attend
0	Monday	34965.666667
1	Wednesday	37585.166667
2	Friday	40116.923077
3	Thursday	40407.400000
4	Sunday	42268.846154
5	Saturday	43072.923077
6	Tuesday	47741.230769

In [13]:
```
# Using explode to highlight the last 2 groups
explode=[0,0,0,0,0,0.1,0.1]
# Creating Pie chart of attendance with percentage enabled.
plt.pie(dodgers_group_df1["attend"],labels=dodgers_group_df1["day_of_week"],explode=explode,shadow=True,
startangle=30,
        autopct="%1.1f%%",
        colors=sns.color_palette("deep"),
        wedgeprops={'edgecolor':'black'})
# setting the title and the  plot sizes
plt.title("Average Attendence by Day of the week")
plt.figure(figsize=(20, 15))
plt.show()
```




```
<Figure size 2000x1500 with 0 Axes>
```

Observations from Pie Plot: The Pieplot indicates that Most people attended the games on Tuesdays followed by Saturdays and Sundays. Mondays had the lowest attendance, probably because of the start of the work week. Hence, ***the Day of the week can be a factor in improving the Attendance***

2.5. Barplot of Game attendance by Promotions

In [14]:

```
def create_promotions_group(colname):
    """
    This function will create a new dataframe with Mean value of attendance for each promotion.
    It takes promotion column and input and return a Dataframe.
    """
    # Creating a dataframe that has data grouped by Promotion and then calculating the average attendance
    dodgers_group_df=dodgers_df.groupby([colname]).mean()['attend'].round().reset_index()
    # Renaming the column name as Yes_No
    dodgers_group_df.rename(columns = {colname:'Yes_No'}, inplace = True)
```

```
    # Adding a new column in the dataframe denoting the Promotion name
    dodgers_group_df["category"]=colname
    # Returning the Dataframe
    return dodgers_group_df
```

In [15]:
```
# Getting the list of promotion columns
promotion_cols=["shirt","cap","fireworks","bobblehead"]
# Creating a new Grouped Dataframe for each Promotion type
for colname in promotion_cols:
     globals()[f"dodgers_group_df_{colname}"]=create_promotions_group(colname)
```

In [16]:
```
dodgers_group_df_bobblehead
```

Out[16]:

	Yes_No	attend	category
0	NO	39138.0	bobblehead
1	YES	53145.0	bobblehead

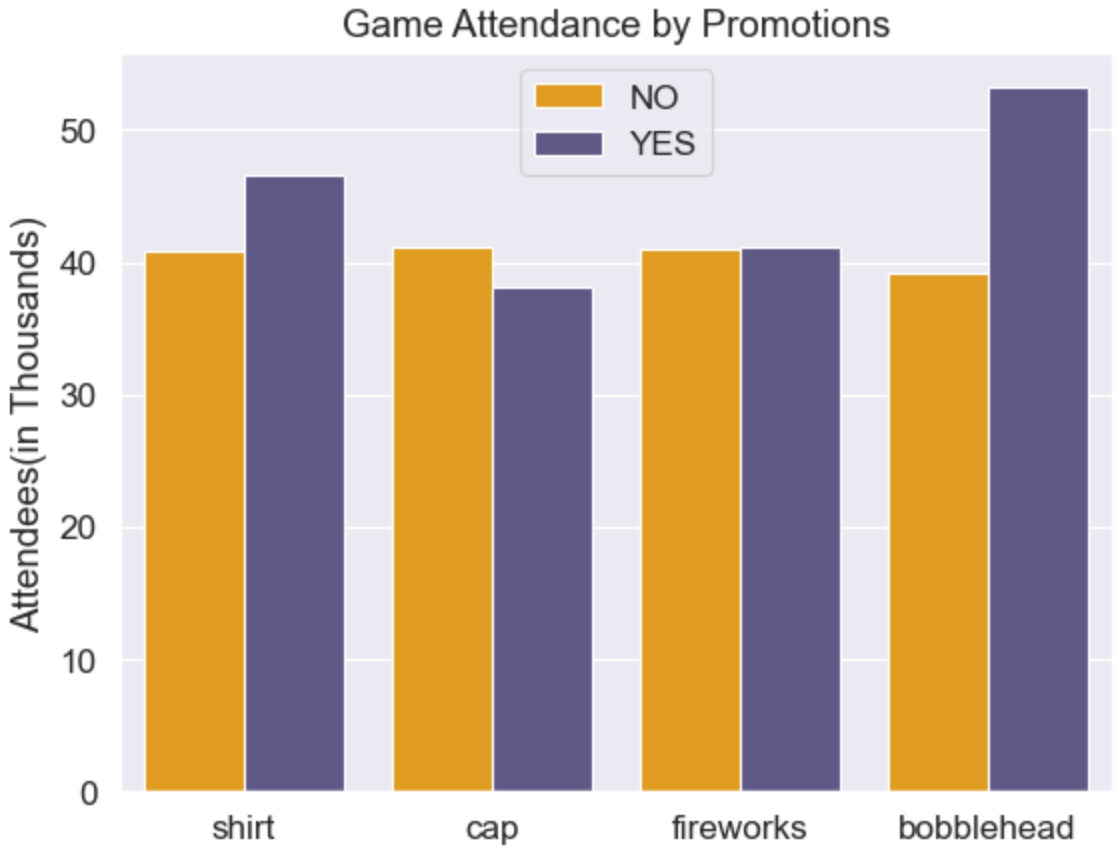
In [17]:
```
# Combining the data from each grouped Dataframe
concat_df1=pd.concat([dodgers_group_df_shirt,
dodgers_group_df_cap,dodgers_group_df_fireworks,dodgers_group_df_bobblehead], ignore_index=True, axis=0)
# Representing attendance in thousands
concat_df1['attend']=concat_df1['attend']/1000
concat_df1
```

Out[17]:

	Yes_No	attend	category
0	NO	40.825	shirt
1	YES	46.644	shirt
2	NO	41.112	cap
3	YES	38.190	cap
4	NO	41.032	fireworks
5	YES	41.078	fireworks
6	NO	39.138	bobblehead
7	YES	53.145	bobblehead

In [18]:

```
# Creating a barplot for each type pf Promotion with Yes or No category
sns.set_style('darkgrid')
sns.set_context('paper', font_scale=1.4)
# Setting colors for Yes and No categories
my_pal2 = {"YES": "#58508d", "NO": "#ffa600"}
# Creating barplot on Promotions vs Attendance and categorized based on Yes or NO values
sns.barplot(x='category',y='attend',hue="Yes_No",data=concat_df1, estimator=np.median,palette=my_pal2)
# Setting the Title and labels
plt.title('Game Attendance by Promotions')
plt.xlabel('')
plt.ylabel('Attendees(in Thousands)')
plt.legend(title="")
plt.show()
```



Observations from Bar Plot: The Promotions plot indicates that Giving away bobbleheads had a great effect on the game attendance. The attendance improved by almost 15,000 on the games when Bobbleheads were given away. Fireworks had no effect on the Game attendance. Giving away T-shirt helped improve the audience, while giving away Caps had a negative effect.Hence overall, ***Bobblehead and Tshirts have a better effect in improving attendance compared to Cap or Fireworks***

2.6. Barplot of Game attendance by Weather and Time of the day

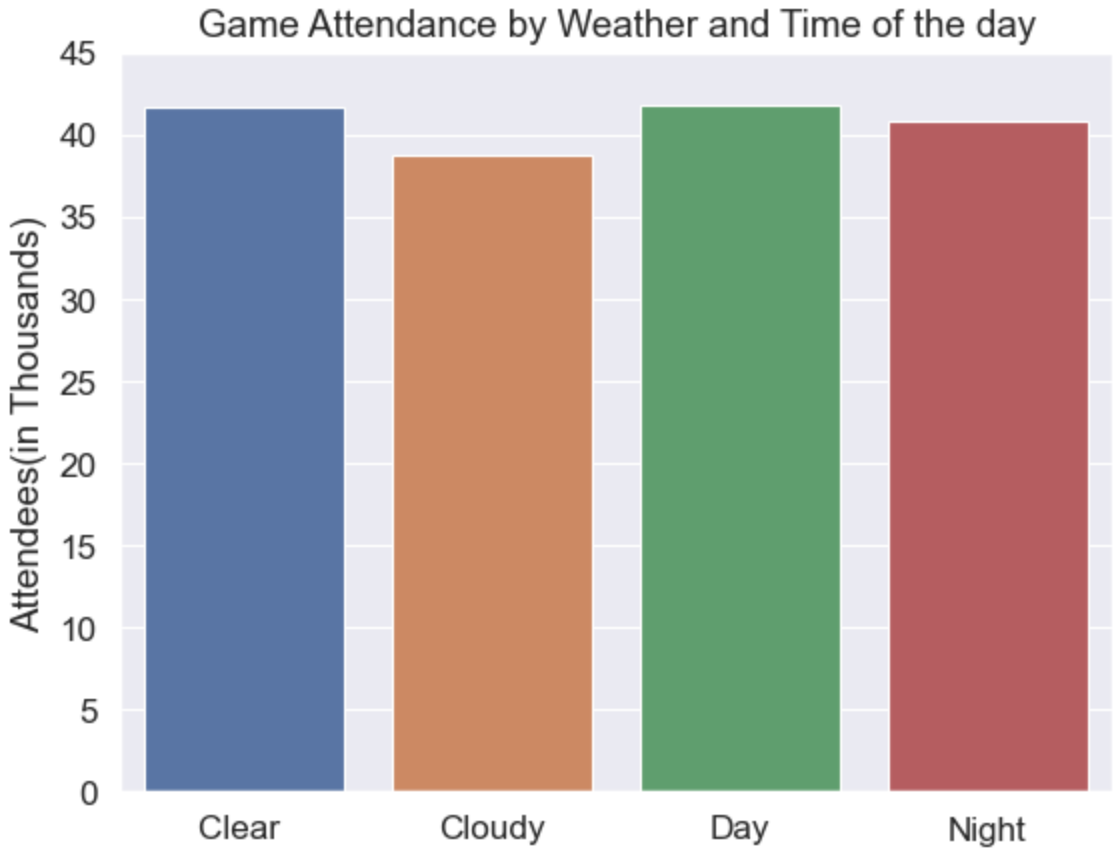
In [19]:
```
# Creating a dataframe that has data grouped by skies and then calculating the average attendance
dodgers_group_df6=dodgers_df.groupby(['skies']).mean()['attend'].round().reset_index()
# Renaming the column name as Condition
dodgers_group_df6.rename(columns = {'skies':'condition'}, inplace = True)
# Creating a dataframe that has data grouped by day_night and then calculating the average attendance
dodgers_group_df7=dodgers_df.groupby(['day_night']).mean()['attend'].round().reset_index()
# Renaming the column name as Condition
dodgers_group_df7.rename(columns = {'day_night':'condition'}, inplace = True)
```

In [20]:
```
# Concatenating the Dataframe of Weather and Skies for visualizations
concat_df2=pd.concat([dodgers_group_df6,dodgers_group_df7], ignore_index=True, axis=0)
# Representing attendance in thousands
concat_df2['attend']=concat_df2['attend']/1000
concat_df2
```

Out[20]:

	condition	attend
0	Clear	41.729
1	Cloudy	38.791
2	Day	41.793
3	Night	40.869

In [21]:
```
# Creating a Barplot attendance vs Weather and Sky conditions
sns.barplot(x='condition',y='attend',data=concat_df2, estimator=np.mean)
# Setting the title and labels
plt.title('Game Attendance by Weather and Time of the day')
plt.xlabel('')
plt.ylabel('Attendees(in Thousands)')
plt.ylim(0, 45)
plt.show()
```



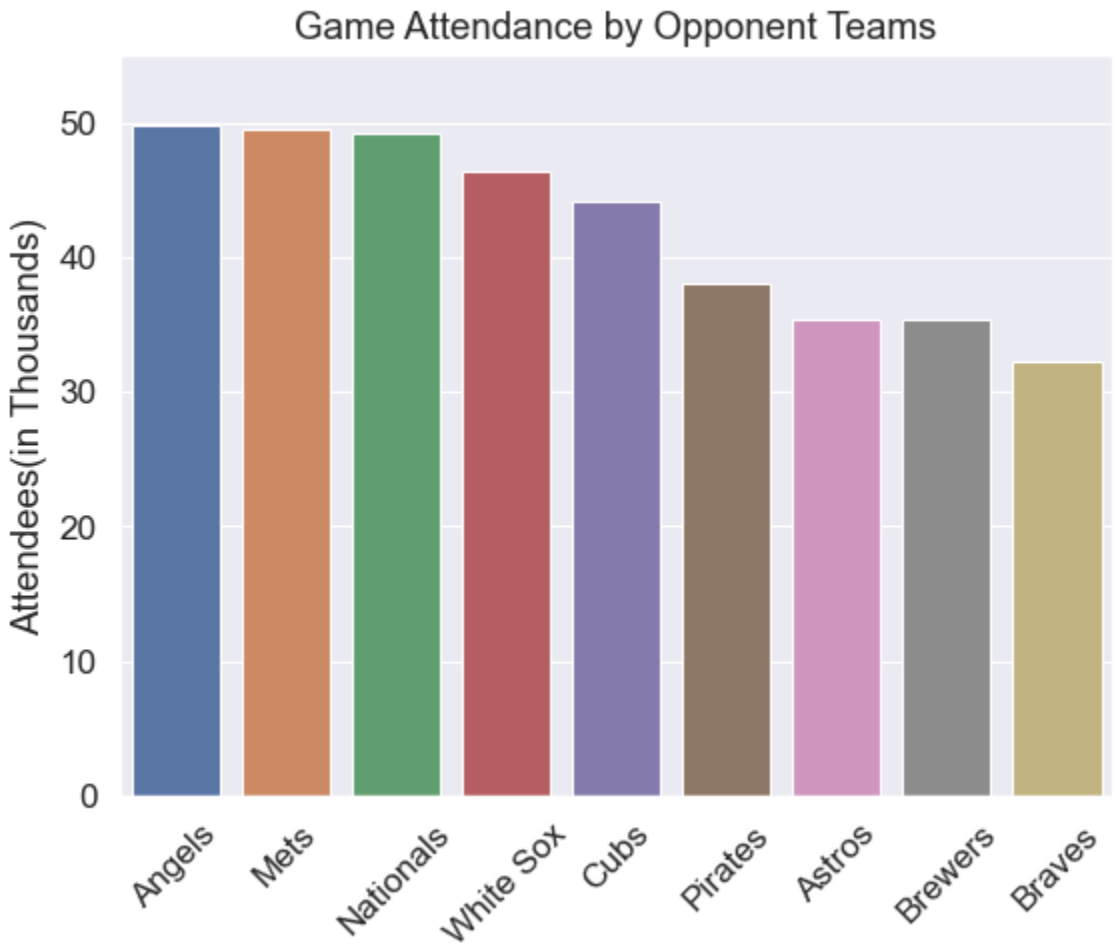
Observations from Bar Plot: The plot indicates that More people attended the games during Clear weather conditions than Cloudy days. Also, More people on average attended the Games during the day than the Night games.

2.7.Barplot of Game attendance by the opponent team

In [22]:
```
# Creating a Grouped Dataframe by Opponents and calculating the mean of attendance
dodgers_group_df8=dodgers_df.groupby('opponent').mean()['attend'].round().reset_index().sort_values(by=
['attend'],ascending=False)
# Printing the top 5 opponents grouped by Game attendance
dodgers_group_df8["attend"]=dodgers_group_df8["attend"]/1000
concat3_df=pd.concat([dodgers_group_df8[:5], dodgers_group_df8[-4:]], ignore_index=True, axis=0)
```

In [23]:
```
# Creating a Barplot attendance vs Weather and Sky conditions
sns.barplot(x='opponent',y='attend',data=concat3_df, estimator=np.mean)
```

```
# Setting the title and labels
plt.title('Game Attendance by Opponent Teams')
#plt.figure(figsize=(10,10))
plt.xlabel('')
plt.ylabel('Attendees(in Thousands)')
plt.xticks(rotation=45)
plt.ylim(0, 55)
plt.show()
```



Observations from Bar Plot: The plot indicates that Most people attended the games of Dodgers vs Los Angeles Angels, New York Mets and Washington Nationals. While Dodgers vs Atlanta Braves had the lowest attendance. Hence, opponents can be a factor for improving the attendance.

2.8. Line plot of Game attendance on Tuesdays and Wednesdays by opponent team

In [24]:
```
# Creating a dataframe of Games held on Tuesdays and wednesdays
tuesday_df=dodgers_df[dodgers_df['day_of_week'].isin(['Tuesday'])]
wednesday_df=dodgers_df[dodgers_df['day_of_week'].isin(['Wednesday'])]
tuesday_df.head()
```

Out[24]:

	month	day	attend	day_of_week	opponent	temp	skies	day_night	cap	shirt	fireworks	bobblehead
0	APR	10	56000	Tuesday	Pirates	67	Clear	Day	NO	NO	NO	NO
7	APR	24	44014	Tuesday	Braves	63	Cloudy	Night	NO	NO	NO	NO
13	MAY	8	32799	Tuesday	Giants	75	Clear	Night	NO	NO	NO	NO
19	MAY	15	47077	Tuesday	Snakes	70	Clear	Night	NO	NO	NO	YES
27	MAY	29	51137	Tuesday	Brewers	74	Clear	Night	NO	NO	NO	YES

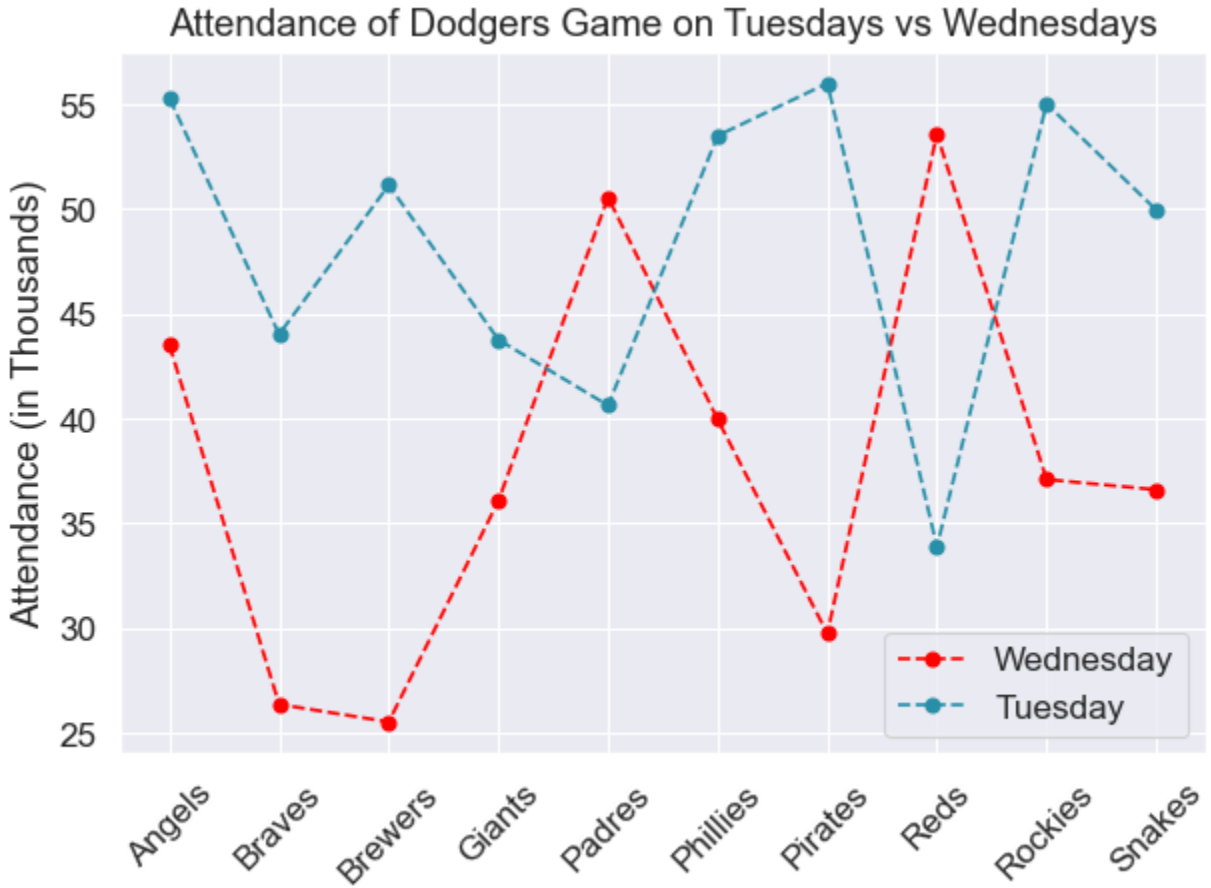
In [25]:
```
# Grouping the data by Opponent and computing the mean attendance
tues_group1_df=tuesday_df.groupby(['opponent']).mean()['attend'].reset_index().sort_values(by=["opponent"])
wed_group1_df=wednesday_df.groupby(['opponent']).mean()['attend'].reset_index().sort_values(by=["opponent"])
tues_group1_df.head()
```

Out[25]:

	opponent	attend
0	Angels	55279.000000
1	Braves	44014.000000
2	Brewers	51137.000000
3	Giants	43757.333333
4	Padres	40619.000000

In [26]:
```
# Plotting the line plot of Tuesday's Game attendance vs Wednesday
plt.plot(wed_group1_df["opponent"],wed_group1_df["attend"]/1000,marker="o",color="Red",linestyle='--
',label="Wednesday")
plt.plot(tues_group1_df["opponent"],tues_group1_df["attend"]/1000,linestyle='--
',marker="o",color="#278FA8",label="Tuesday")
```

```
# Setting the title and labels of line plot
plt.xticks(rotation=45)
plt.title("Attendance of Dodgers Game on Tuesdays vs Wednesdays")
plt.ylabel("Attendance (in Thousands)")
plt.legend()
plt.tight_layout()
```



Observations from Line Plot: The Lineplot indicates that overall more people attended the Dodgers games on Tuesdays compared to wednesdays, with the exception of SanDiego Padres and Cincinnati Reds.

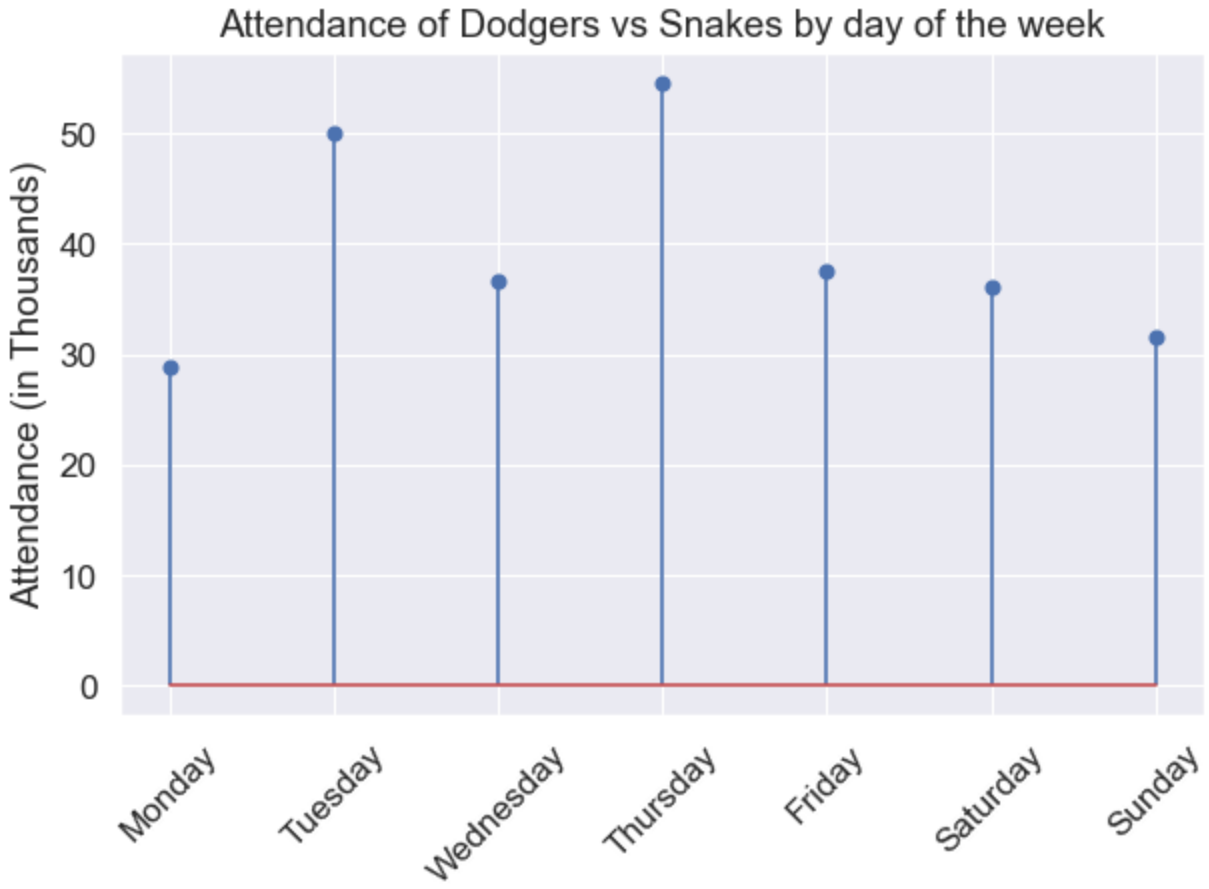
2.9. Stem plot of Dodgers vs Snakes attendance by day of the week

In [27]:
```
# Creating a stemplot of Snakes Game by the day  of the week
```

```
dodgers_group_df10=dodgers_df[dodgers_df["opponent"].isin(['Snakes'])].groupby(['opponent','day_of_week']).mean()
['attend'].reset_index()
# Assigning Numbers to the Day of the week
dodgers_group_df10['day_of_week'] = dodgers_group_df10['day_of_week'].map({
     'Monday':1,
     'Tuesday':2,
     'Wednesday':3,
    'Thursday':4,
    'Friday':5,
     'Saturday':6,
     'Sunday':7
})
# Sorting by the day of the week
dodgers_group_df10=dodgers_group_df10.sort_values(by=['day_of_week'])
```

In [28]:

```
# Creating list of months that will be used as labels in the line plot
dow=['Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday']
# Plotting stem plot of Day of the week vs Game attendance
plt.stem(dow,dodgers_group_df10["attend"]/1000)
# Setting the title and labels
plt.xticks(rotation=45)
plt.title("Attendance of Dodgers vs Snakes by day of the week")
plt.ylabel("Attendance (in Thousands)")
plt.tight_layout()
```



Observations from Stem Plot: The Stemplot indicates that more people attended the Snakes game on Thursdays and Tuesdays. Mondays had the least attendance. The common trend that can be seen from most of these plots is that, Mondays had the least attendance, so the Day of the week is a factor in improving the attendance.

In [29]:
```
# Making a copy of dodgers Dataframe
dodgers_df1=dodgers_df.copy()
# Using Map function to assign numbers to Promotion columns
for col in promotion_cols:
    # Looping through each column and assigning 0 to No and 1 for Yes
    dodgers_df1[col] = dodgers_df1[col].map({
    'NO':0,
    'YES':1
    })
```

In [30]:
```
# Creating a dataframe with subset of Promotion columns with the Day of the week
dodgers_group_df11=dodgers_df1[["day_of_week","cap","shirt","fireworks","bobblehead"]]
dodgers_group_df11.head()
```

Out[30]:

	day_of_week	cap	shirt	fireworks	bobblehead
0	Tuesday	0	0	0	0
1	Wednesday	0	0	0	0
2	Thursday	0	0	0	0
3	Friday	0	0	1	0
4	Saturday	0	0	0	0

In [31]:
```
# Grouping the data by Day of the week and calculing the sum of Promotions
dodgers_group_df12=dodgers_group_df11.groupby(['day_of_week']).sum().reset_index()
# Using Map function to assign numbers to Day of the week
dodgers_group_df12['day_of_week'] = dodgers_group_df12['day_of_week'].map({
     'Monday':1,
     'Tuesday':2,
     'Wednesday':3,
    'Thursday':4,
    'Friday':5,
     'Saturday':6,
     'Sunday':7
})
# Sorting the dataframe by the day of the week
dodgers_group_df12=dodgers_group_df12.sort_values(by=["day_of_week"])
dodgers_group_df12
```

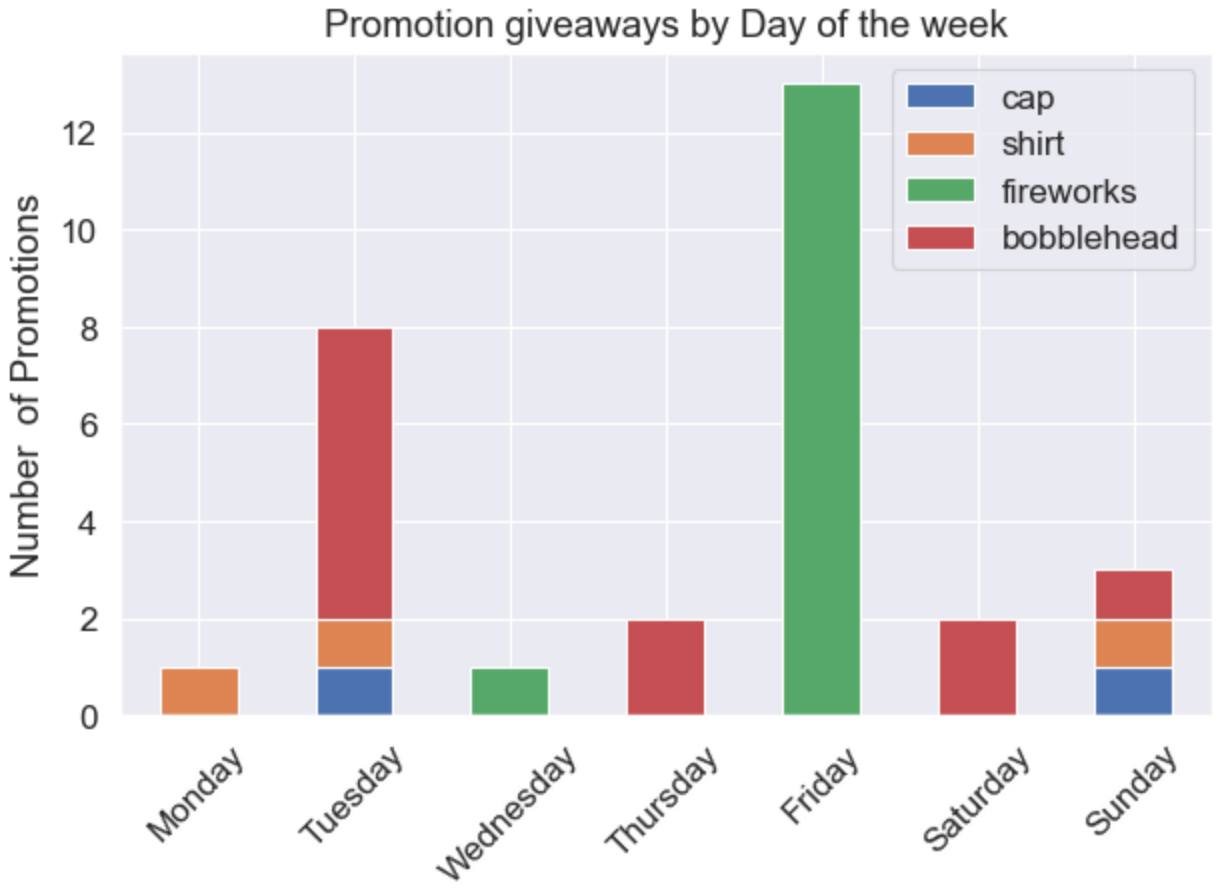
Out[31]:

	day_of_week	cap	shirt	fireworks	bobblehead
1	1	0	1	0	0
5	2	1	1	0	6
6	3	0	0	1	0
4	4	0	0	0	2
0	5	0	0	13	0
2	6	0	0	0	2
3	7	1	1	0	1

2.10. Stacked Barplot of Promotions by Day of the week

In [32]:

```
# Setting the Day of the week values as X index for plotting in a Barplot
x_indexes=dodgers_group_df12['day_of_week'].values-1
# Plotting Stacked barplot of Number of Promotions
dodgers_group_df12.plot(x='day_of_week', kind='bar', stacked=True)
# Setting the xticks with Day of the week
plt.xticks(ticks=x_indexes,labels=dow)
# Setting the Title, X and Y labels
plt.legend()
plt.xticks(rotation=45)
plt.title("Promotion giveaways by Day of the week")
plt.ylabel("Number  of Promotions")
plt.xlabel("")
plt.tight_layout()
plt.show()
```

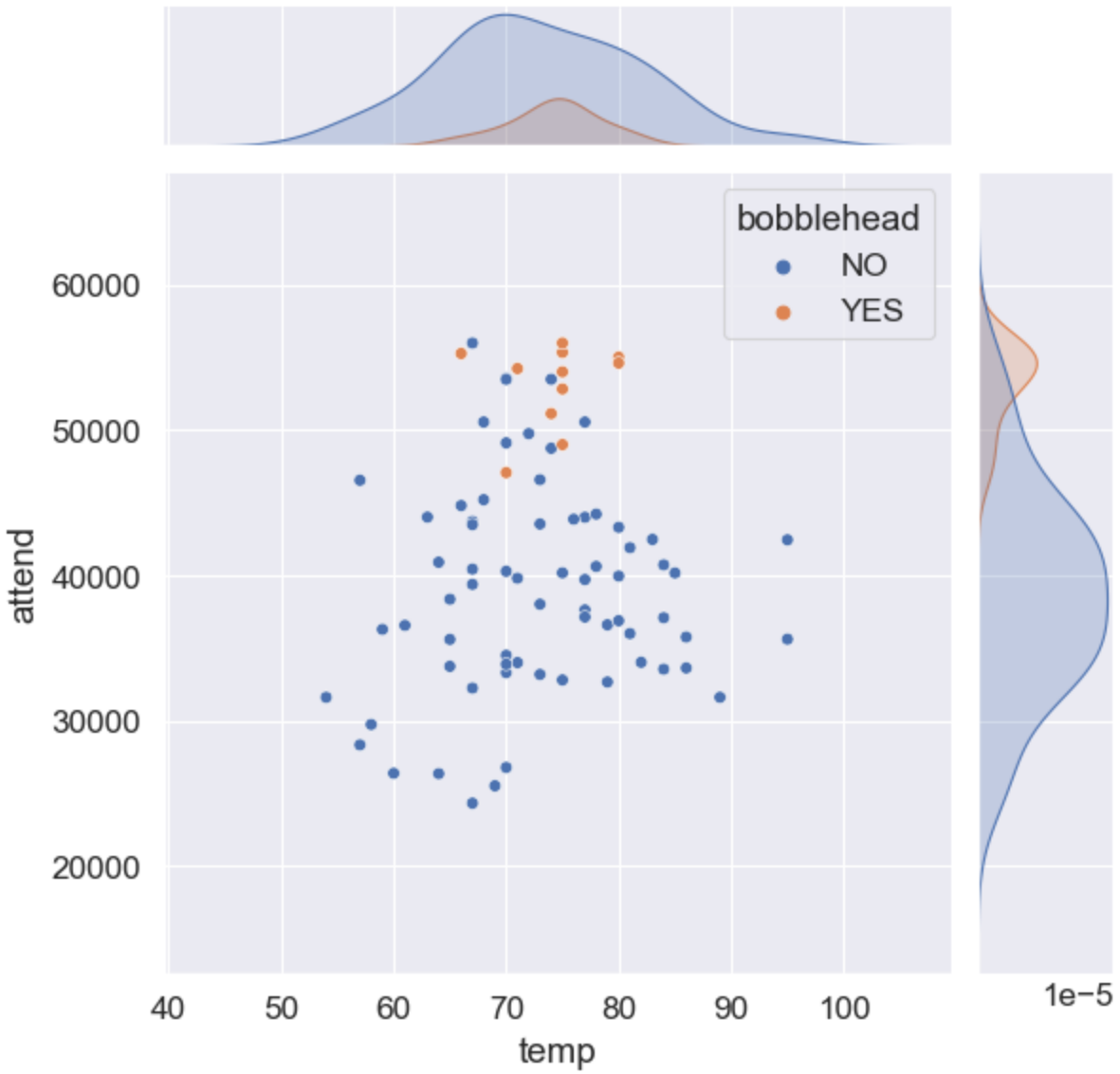


Observations from Stacked BarPlot: The plot indicates that Fridays had the most number of Promotions and Fireworks were used mostly on Fridays. Bobbleheads were given away mostly on Tuesdays. Mondays and Wednesdays had the least number of Promotions. The bobblehead promotions given on Tuesdays may be a factor in improving the attendance on Tuesdays.

2.11. Jointplot of Temperature vs Attendance categorized by Bobblehead

In [33]:
```
# Creating Jointplot of Temperature vs Attendance categorized by Bobblehead
sns.jointplot(data=dodgers_df, x="temp", y="attend", hue="bobblehead")
```

Out[33]:
```
<seaborn.axisgrid.JointGrid at 0x2275a755040>
```

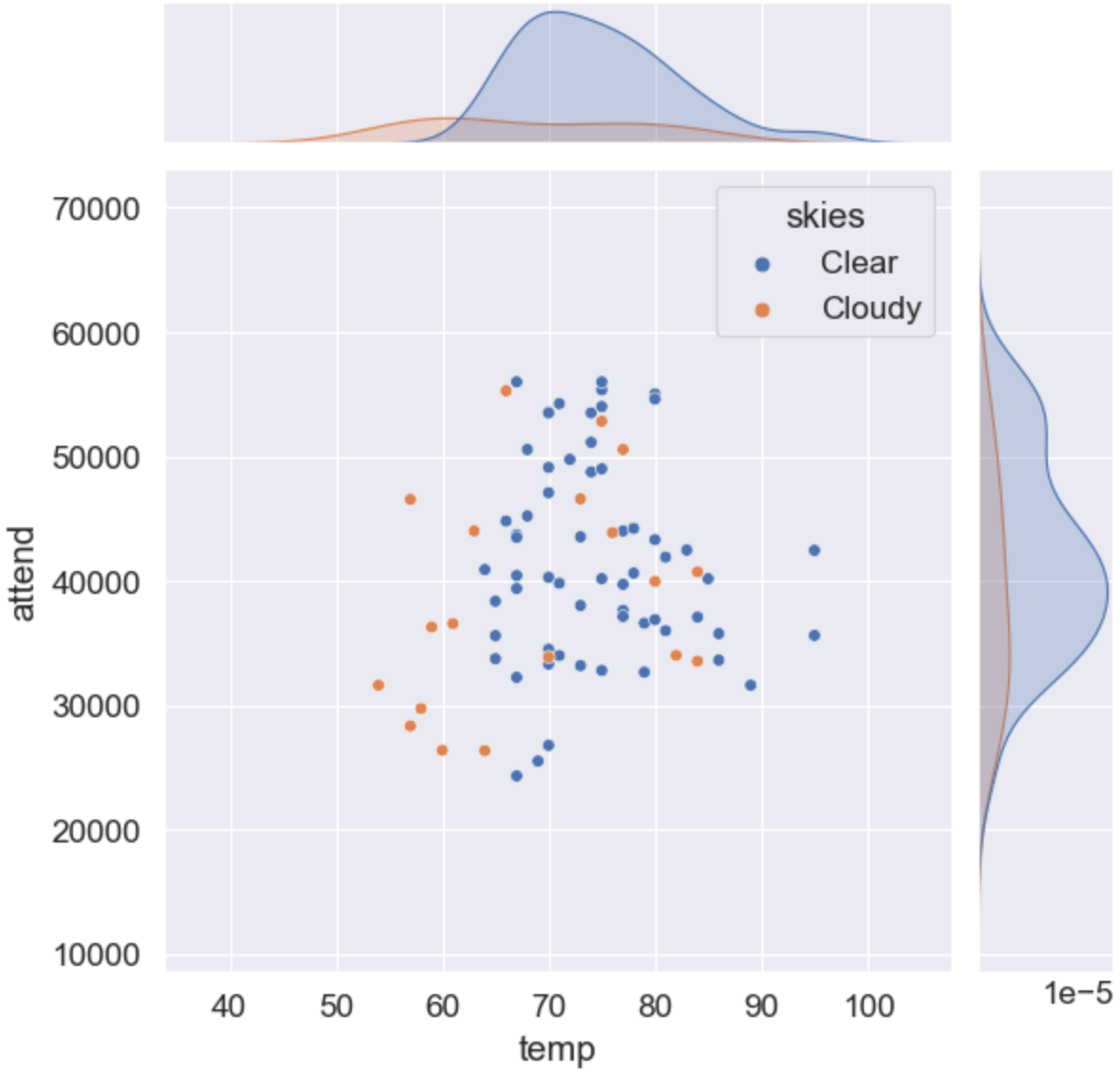


Observations from JointPlot1: The plot indicates that most bobblehead giveaways were on the days when temperature was between 65 and 80 degrees, when the game attendance was the highest. ***This indicates that Bobblehead may be a big contributor for improving the game attendance***

2.11. Jointplot of Temperature vs Attendance categorized by weather condition

In [34]:
```
# Creating Jointplot of Temperature vs Attendance categorized by Sky condition
sns.jointplot(data=dodgers_df, x="temp", y="attend", hue="skies")
```

Out[34]: <seaborn.axisgrid.JointGrid at 0x2275ab9c430>

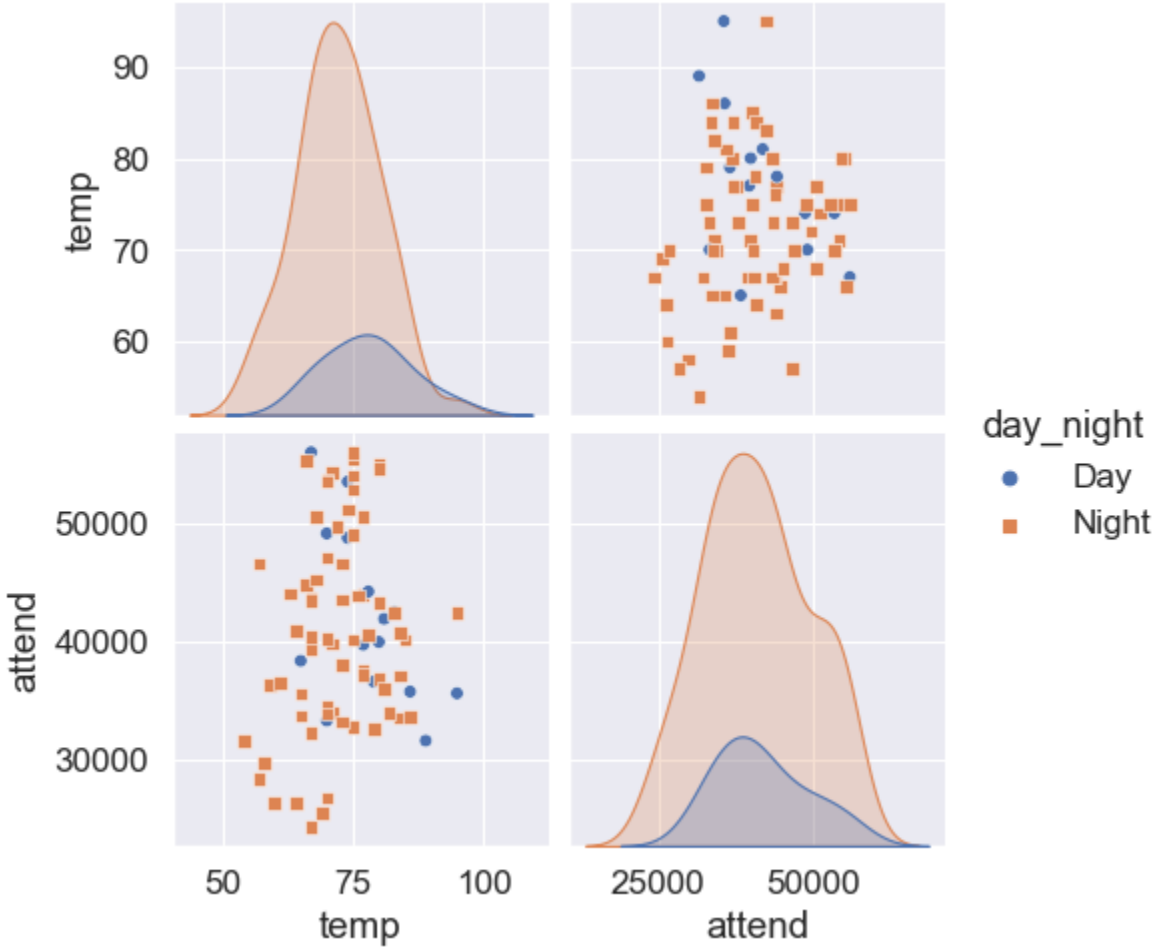


Observations from JointPlot2: The plot indicates that the temperature was cooler (less than 65 degrees) during most of the cloudy days that had lower attendance. Most people attended the games when the sky conditions were clear.

2.12. Pairplot

In [35]:
```
# Getting the list of columns to be used in Pairplot
pair_cols=["temp","attend","skies","day_night"]
plt.figure(figsize=(18,18))
# Plotting the pairplot based on the Day night condition
sns.pairplot(data=dodgers_df1[pair_cols],hue="day_night",markers=["o", "s"])
plt.show()
```

```
<Figure size 1800x1800 with 0 Axes>
```



Observations from Pairplot: The Pairplot of Temperature vs Attendance based on the day night condition indicates that most of

the games were scheduled during the Night. The Night games when temperature was cooler had fewer people attending the game. The game attendance improved during warmer nights.

2.13. Heatmap1

In [36]:
```
# Creating a subset of dataframe by grouping by month, day of the week
dodgers_group_df13=dodgers_df1[['month','day_of_week','attend']].groupby(['month','day_of_week']).mean().reset_index(
# Using Map function to assign numbers to Day of the week
dodgers_group_df13['day_of_week'] = dodgers_group_df13['day_of_week'].map({
     'Monday':1,
     'Tuesday':2,
     'Wednesday':3,
    'Thursday':4,
    'Friday':5,
     'Saturday':6,
     'Sunday':7
})
# Using Map function to assign numbers to Month
dodgers_group_df13['month'] =dodgers_group_df13['month'].map({
   'APR':4,
    'MAY':5,
    'JUN':6,
    'JUL':7,
    'AUG':8,
    'SEP':9,
    'OCT':10
})
dodgers_group_df13=dodgers_group_df13.sort_values(by=["month","day_of_week"])
```

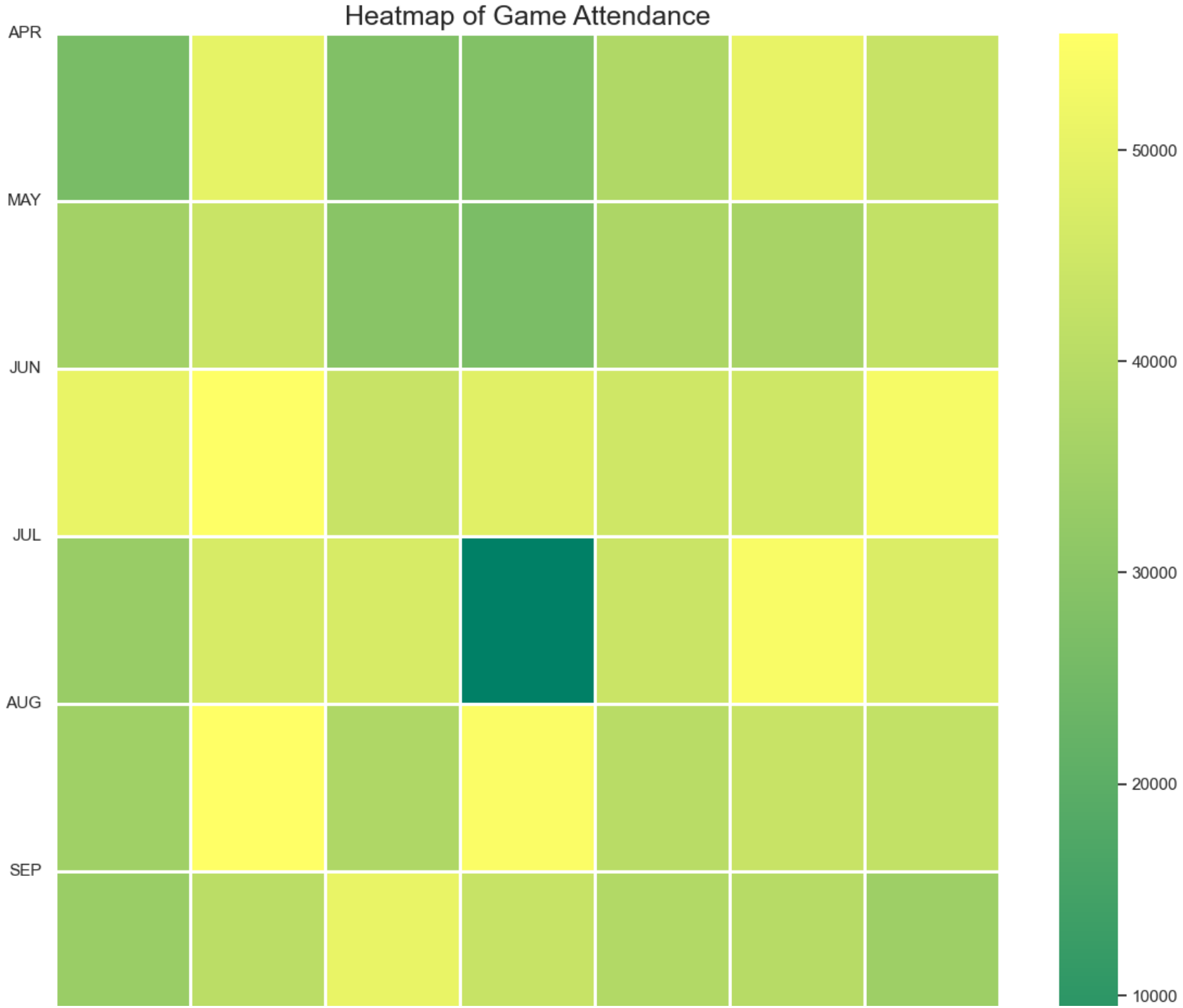
In [37]:
```
# Assigning values to Month names and indexes to be used while plotting
month_names=["APR","MAY","JUN","JUL","AUG","SEP","OCT"]
y_indexes=np.arange(len(month_names))
x_indexes=np.arange(len(dow))
# Creating a pivot table of Division by Crash month
pivot1=dodgers_group_df13.pivot_table(index='month',columns='day_of_week',values='attend').fillna(0)
pivot1
```

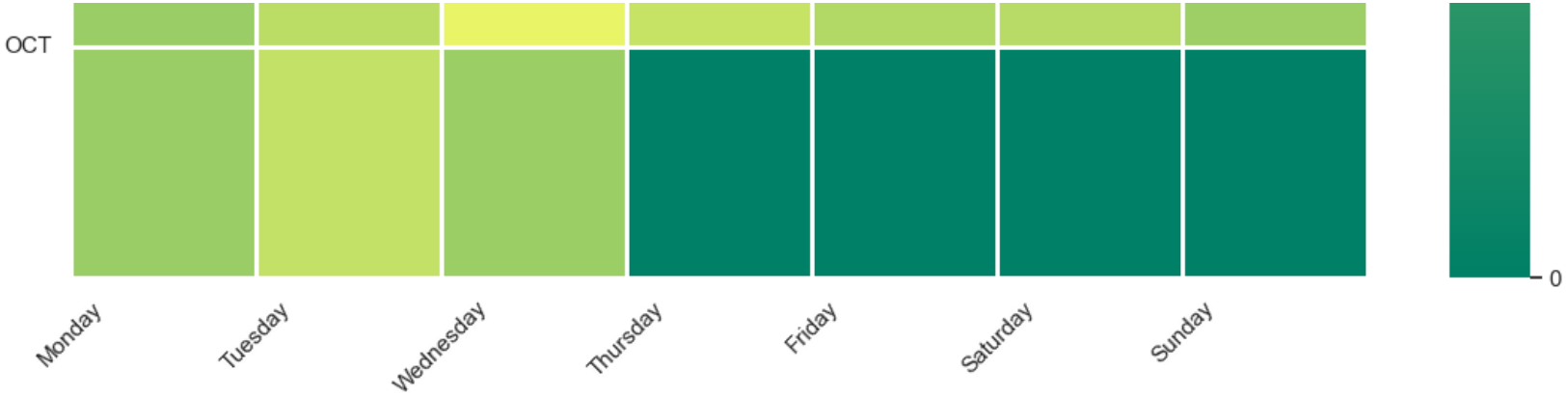
Out[37]:

day_of_week	1	2	3	4	5	6	7
month							
4	26376.000000	50007.0	28037.0	28328.0	38204.000000	50395.500000	43556.000000
5	35347.000000	43671.0	29751.0	26773.0	37593.333333	36559.666667	42145.000000
6	50559.000000	55279.0	43494.0	49006.0	45097.500000	44713.500000	53504.000000
7	33303.666667	46738.0	46762.5	0.0	43873.000000	54014.000000	47537.000000
8	34768.500000	55512.0	37951.0	54621.0	40321.333333	43436.000000	42201.000000
9	33540.000000	40619.0	50560.0	43309.0	38650.000000	39721.666667	34322.666667
10	33624.000000	42473.0	34014.0	0.0	0.000000	0.000000	0.000000

In [46]:

```
# Creating HeatMap of Attendance by Day of the week and by Month
sns.heatmap(pivot1,cmap='summer',linecolor='white',linewidth=1)
# Assigning title and labels of the HeatMap
plt.title('Heatmap of Game Attendance',fontsize = 18)
plt.yticks(ticks=y_indexes,labels=month_names)
plt.xticks(ticks=x_indexes,labels=dow)
plt.xticks(rotation=45)
plt.yticks(rotation=0)
plt.ylabel('')
plt.xlabel('')
plt.show()
```


Heatmap of Game Attendance
APR
MAY
JUN
JUL
AUG
SEP
50000
40000
30000
20000
10000

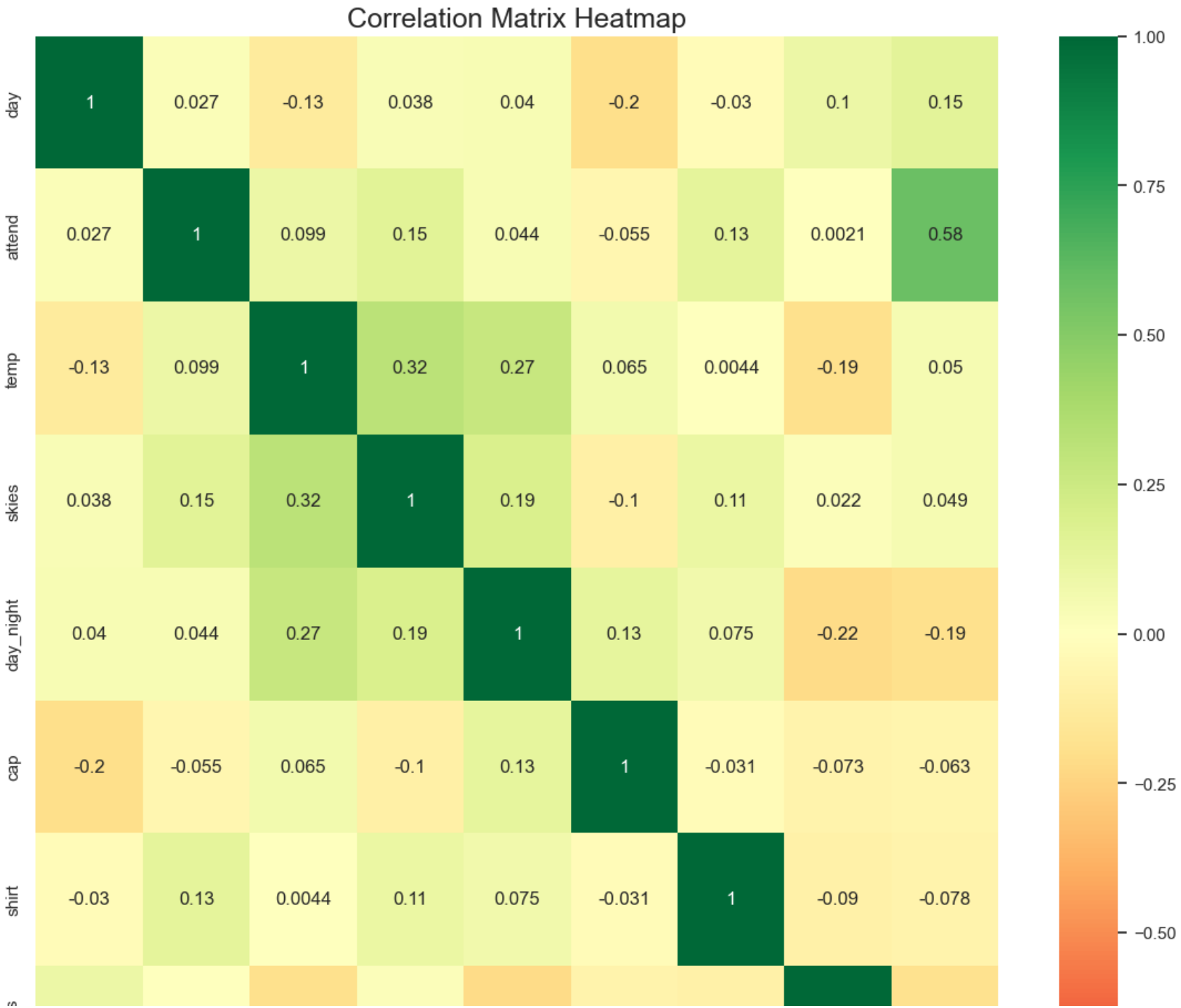


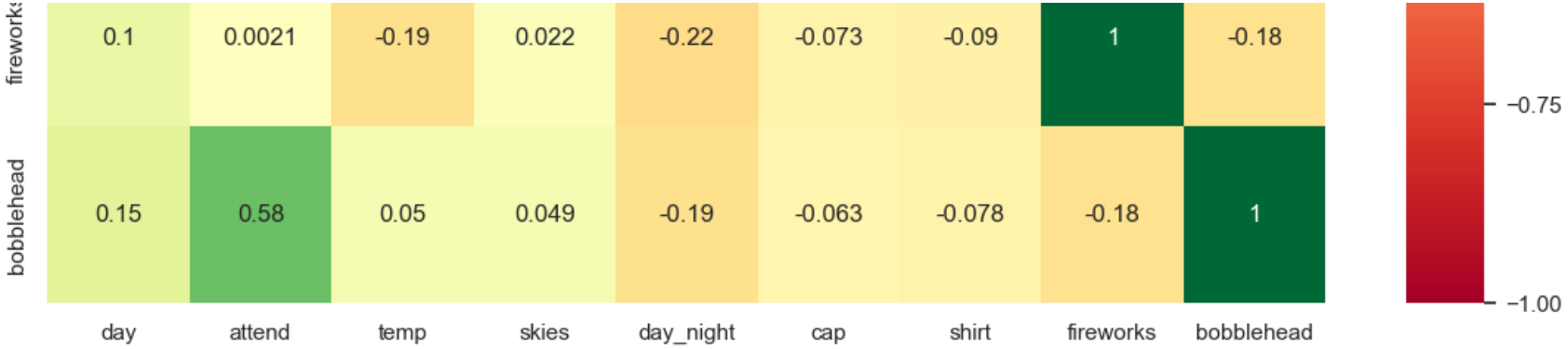
Observations from HeatMap: The heatmap indicates that Tuesdays in the months of April,June, July and August were very popular. More people also attended the games on Saturdays and sundays during these months.

2.14. Heatmap2 of Correlation Matrix

In [39]:
```
# Using Map function to assign numbers to skies and day_night columns
dodgers_df1["skies"] = dodgers_df1["skies"].map({
    'Clear ':1,
    'Cloudy':0
    })
dodgers_df1["day_night"] = dodgers_df1["day_night"].map({
    'Day':1,
    'Night':0
    })
# Creating Correlation Matrix on the Dodgers Data
dodgers_corr=dodgers_df1.corr()
```

In [45]:
```
# Creating a HeatMap using Seaborn with annotation turned on
sns.heatmap(dodgers_corr, annot=True, vmax=1, vmin=-1, center=0,cmap ='RdYlGn')
sns.set(font_scale=1)
# setting the size of the plot
sns.set(rc={"figure.figsize": (14, 14)})
plt.title('Correlation Matrix Heatmap',fontsize = 18)
plt.show()
```


Correlation Matrix Heatmap
day
attend
temp
skies
day_night
cap
shirt
1 0.027 -0.13 0.038 0.04 -0.2 -0.03 0.1 0.15
0.027 1 0.099 0.15 0.044 -0.055 0.13 0.0021 0.58
-0.13 0.099 1 0.32 0.27 0.065 0.0044 -0.19 0.05
0.038 0.15 0.32 1 0.19 -0.1 0.11 0.022 0.049
0.04 0.044 0.27 0.19 1 0.13 0.075 -0.22 -0.19
-0.2 -0.055 0.065 -0.1 0.13 1 -0.031 -0.073 -0.063
-0.03 0.13 0.0044 0.11 0.075 -0.031 1 -0.09 -0.078
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50



Observations from Correlation Matrix HeatMap: The heatmap indicates that bobblehead is positively correlated with the attendance and has greater chances of improving the attendance. T-shirt and Sky conditions also seem to have positive correlation with attendance. These results indicate that ***Bobbleheads and T-shirt promotions given on the Game days when sky condition is clear, can have better chances of improving the attendance***

3. Building a Linear Regression Model using OLS method in statsmodels package

In [41]:

```
# Creating a function to try the OLS method on each variable and returns the R squared value
def create_Mining(dodgers_df1,fieldname):
    """Searches for each variable in the dataframe and performs ordinary least squares test.
    dodgers_df1: Source dataframe in which each variables is tested.
    fieldname: The dependent variable.
    returns: list of (rsquared, variable name) pairs
    """
    # creating a list to store all the R2 results of each variable
    all_fields_R2 = []
    # Iterate through each column in the source dataframe
    for name in dodgers_df1.columns:
        # Using try and except clause to catch exception while calculating variance on non-numeric fields
        try:
            # Excluding columns that have very small variance as they may not be significant.
            if dodgers_df1[name].var() < 1e-7:
                continue
            # Creating formula used in regression. The fieldname is the dependent variable and "name" is
            # the explanatory variable. This will be iterated for each column in the dataframe.
            formula = fieldname +' ~ + ' + name
            # Using statsmodel.formula.api to calculate the ordinary least squares
            model = smf.ols(formula, data=dodgers_df1)
            # if number of observations is lesser than half the length of dataframe ignore the column.
```

```
            # This check is essential to ignore the columns with many NaNs
            if model.nobs < len(dodgers_df1)/2:
                continue
            # This will store the results of the model in results variable
            results = model.fit()
        # To catch exceptions while performing variance on non-numeric columns
        except (ValueError, TypeError, patsy.PatsyError) as e:
            continue
        # After each loop the results of the R2 and the column names are added back to all_fields_R2 list
        all_fields_R2.append((results.rsquared, name))
    return all_fields_R2
```

In [42]:
```
# Calling the Mining function of the dodgers dataframe created in step above.
all_fields_R2 = create_Mining(dodgers_df1,'attend')
# Sort the results so that the top few columns with significant R2 will be considered for analysis
all_fields_R2.sort(reverse=True)
# Pring the top results from all_fields_R2
for rsq,col in all_fields_R2[:10]:
    print(rsq,col)
```

```
1.0 attend
0.3386017539503713 bobblehead
0.022789974703524174 skies
0.01776053877036865 shirt
0.009791247146878956 temp
0.0030252686380446425 cap
0.0018960761828167305 day_night
0.0007340294287303539 day
4.386761970121e-06 fireworks
```

Observations: The results of the r-squared values from the model indicates that out of all the promotions, Bobblehead had a big share of improving the attendance followed by the T-shirt. The Temperature also played a key role and was better than the Cap and Fireworks promotions.

In [43]:
```
def create_ols_models(features,model_results_df):
    """
    This function creates a linear regression model based using the Ordinary Least Squares method
    and the fields passed as arguments.
    It returns r-squared,adjusted r-squared values along with p-values and the Intercept.
    """
    # Calculating the formula to be used in the ols model
    formula="attend ~ "+ features
```

```
    # Creating the model and fitting it
    model1 = smf.ols(formula, data=dodgers_df1).fit()
    # Appending the results to the dataframe for display
    model_results_df=model_results_df.append({'columns' : features, 'rsquared' : model1.rsquared, 'rsquared_adj' :
model1.rsquared_adj,
                          'pvalues' : model1.pvalues["bobblehead"],'effect' : model1.params["bobblehead"]
                                      },ignore_index = True)
    return model_results_df
```

In [44]:

```
# Creating an empty Dataframe for displaying the results
model_results_df = pd.DataFrame(columns=['columns','rsquared', 'rsquared_adj', 'pvalues','effect'])
# Creating list of feature combinations to be used for creating the model
features_list=['bobblehead','bobblehead+shirt','bobblehead+cap',
               'bobblehead+fireworks','bobblehead+shirt+cap','bobblehead+shirt+fireworks',
                'bobblehead+shirt+skies+temp',
                'bobblehead+shirt+fireworks+cap','bobblehead+shirt+fireworks+day_of_week+temp',
                'bobblehead+fireworks+opponent+skies+temp',
                'bobblehead+shirt+cap+fireworks+day_of_week+temp+opponent'
               ]
# Looping through the features in the list and building the model and storing the results
for features in features_list:
    model_results_df=create_ols_models(features,model_results_df)
# Displaying the results of the Models created using statsmodel OLS method
model_results_df
```

Out[44]:

	columns	rsquared	rsquared_adj	pvalues	effect
0	bobblehead	0.338602	0.330230	1.216964e-08	14006.707792
1	bobblehead+shirt	0.370660	0.354523	4.242514e-09	14342.785617
2	bobblehead+cap	0.338938	0.321988	1.664887e-08	13978.812834
3	bobblehead+fireworks	0.350559	0.333906	7.272889e-09	14491.689935
4	bobblehead+shirt+cap	0.370801	0.346286	5.864320e-09	14323.944056
5	bobblehead+shirt+fireworks	0.387305	0.363434	1.874356e-09	14943.994854
6	bobblehead+shirt+skies+temp	0.382694	0.350204	7.691049e-09	14166.992775
7	bobblehead+shirt+fireworks+cap	0.387305	0.355058	2.635354e-09	14943.557932
8	bobblehead+shirt+fireworks+day_of_week+temp	0.519027	0.450316	3.771279e-07	13118.481210
9	bobblehead+fireworks+opponent+skies+temp	0.566723	0.422298	2.146636e-08	14917.954215
10	bobblehead+shirt+cap+fireworks+day_of_week+temp+opponent	0.681594	0.519388	1.556136e-04	10875.496768

Steps taken to arrive at the conclusion: To arrive at the conclusion, different visualizations were plotted and the results were analyzed. Correlation matrix was also build and the results were analyzed. Also Linear regression model was built using Ordinary Least squares method and the results of the model with different features were compared to see which model resulted in better adjusted-R2 value to improve the attendance.

Conclusion and Recommendations:

The results from the model indicates that, ***Bobblehead promotion*** is a great way to improve the Game attendance. The model results shows that the ***attendance may improve by 14000*** just by giving away the Bobbleheads.

Combining multiple promotions can also improve the attendance. Giving away ***Bobbleheads and T shirts along with fireworks*** had the best results, as per the model with the adjusted r-squared value of .36 and may improve the attendance by about 14,950 people.

Though we don't have control over Temperature, sky conditions and Opponent teams, these factors can also improve the attendance. ***With all favorable conditions along with bobblehead and fireworks promotion, the model suggests that the attendance can be improved by upto 14,940 people.***