In [13]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df = pd.read_csv('data/dodgers.csv')
df
```

Out[2]:

	month	day	attend	day_of_week	opponent	temp	skies	day_night	cap	shirt	fireworks	bob
0	APR	10	56000	Tuesday	Pirates	67	Clear	Day	NO	NO	NO	
1	APR	11	29729	Wednesday	Pirates	58	Cloudy	Night	NO	NO	NO	
2	APR	12	28328	Thursday	Pirates	57	Cloudy	Night	NO	NO	NO	
3	APR	13	31601	Friday	Padres	54	Cloudy	Night	NO	NO	YES	
4	APR	14	46549	Saturday	Padres	57	Cloudy	Night	NO	NO	NO	
...	...	...	...	...	...	...	...	...	...	...	...	
76	SEP	29	40724	Saturday	Rockies	84	Cloudy	Night	NO	NO	NO	
77	SEP	30	35607	Sunday	Rockies	95	Clear	Day	NO	NO	NO	
78	OCT	1	33624	Monday	Giants	86	Clear	Night	NO	NO	NO	
79	OCT	2	42473	Tuesday	Giants	83	Clear	Night	NO	NO	NO	
80	OCT	3	34014	Wednesday	Giants	82	Cloudy	Night	NO	NO	NO	

81 rows × 12 columns

In [5]:
```
df.dtypes
```

Out[5]:
```
month           object
day              int64
attend           int64
day_of_week     object
opponent        object
temp             int64
skies           object
day_night       object
cap             object
shirt           object
fireworks       object
bobblehead      object
dtype: object
```

In [10]:
```
df['day_of_week']
```

Out[10]:
```
0       Tuesday
1     Wednesday
2      Thursday
3        Friday
```

```
4      Saturday
        ...
76     Saturday
77       Sunday
78       Monday
79      Tuesday
80    Wednesday
Name: day_of_week, Length: 81, dtype: object
```

In [11]:

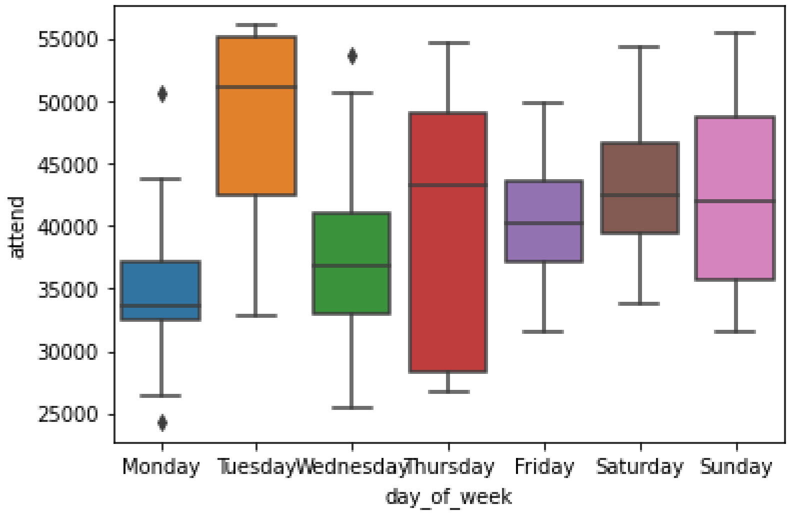
```
#reorder the day of week
df['day_of_week'] = pd.Categorical(df['day_of_week'], categories=
    ['Monday','Tuesday','Wednesday','Thursday','Friday','Saturday', 'Sunday'],
    ordered=True)
```

In [14]:

```
#boxplot attendance by day of the week
sns.boxplot(x = 'day_of_week', y = 'attend', data = df)
```

Out[14]:

```
<AxesSubplot:xlabel='day_of_week', ylabel='attend'>
```

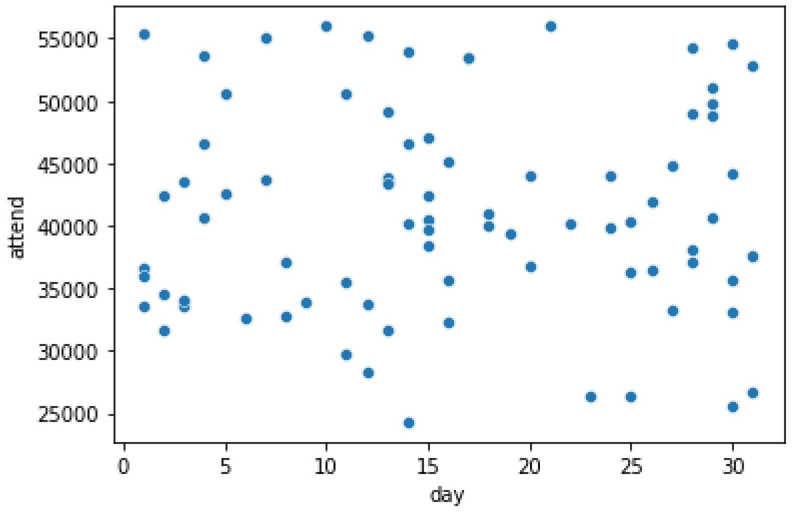


In [17]:

```
#scatterplot attendacne by day of the month
sns.scatterplot(data=df, x="day", y="attend")
```

Out[17]:

```
<AxesSubplot:xlabel='day', ylabel='attend'>
```



In [19]:
```
#get dummies for the days of thw week for model building
weekdays = pd.get_dummies(df['day_of_week'])
weekdays
```

Out[19]:

	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
0	0	1	0	0	0	0	0
1	0	0	1	0	0	0	0
2	0	0	0	1	0	0	0
3	0	0	0	0	1	0	0
4	0	0	0	0	0	1	0
...	...	...	...	...	...	...	...
76	0	0	0	0	0	1	0
77	0	0	0	0	0	0	1
78	1	0	0	0	0	0	0
79	0	1	0	0	0	0	0
80	0	0	1	0	0	0	0

81 rows × 7 columns

In [20]:
```
#combine data frames
df = pd.concat([df, weekdays], axis = 1)
df
```

Out[20]:

	month	day	attend	day_of_week	opponent	temp	skies	day_night	cap	shirt	fireworks	bob
0	APR	10	56000	Tuesday	Pirates	67	Clear	Day	NO	NO	NO	
1	APR	11	29729	Wednesday	Pirates	58	Cloudy	Night	NO	NO	NO	
2	APR	12	28328	Thursday	Pirates	57	Cloudy	Night	NO	NO	NO	
3	APR	13	31601	Friday	Padres	54	Cloudy	Night	NO	NO	YES	
4	APR	14	46549	Saturday	Padres	57	Cloudy	Night	NO	NO	NO	
...	...	...	...	...	...	...	...	...	...	...	...	
76	SEP	29	40724	Saturday	Rockies	84	Cloudy	Night	NO	NO	NO	
77	SEP	30	35607	Sunday	Rockies	95	Clear	Day	NO	NO	NO	
78	OCT	1	33624	Monday	Giants	86	Clear	Night	NO	NO	NO	
79	OCT	2	42473	Tuesday	Giants	83	Clear	Night	NO	NO	NO	
80	OCT	3	34014	Wednesday	Giants	82	Cloudy	Night	NO	NO	NO	

81 rows × 19 columns

In [24]:

```
#drop useless columns
df.drop(['month', 'day_of_week', 'opponent', 'skies', 'day_night', 'cap', 'shirt', 'fir
df
```

Out[24]:

	day	attend	temp	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
0	10	56000	67	0	1	0	0	0	0	0
1	11	29729	58	0	0	1	0	0	0	0
2	12	28328	57	0	0	0	1	0	0	0
3	13	31601	54	0	0	0	0	1	0	0
4	14	46549	57	0	0	0	0	0	1	0
...	...	...	...	...	...	...	...	...	...	...
76	29	40724	84	0	0	0	0	0	1	0
77	30	35607	95	0	0	0	0	0	0	1
78	1	33624	86	1	0	0	0	0	0	0
79	2	42473	83	0	1	0	0	0	0	0
80	3	34014	82	0	0	1	0	0	0	0

81 rows × 10 columns

In [32]:

```
X = df[['Monday', 'Tuesday', 'Wednesday',
        'Thursday', 'Friday', 'Saturday', 'Sunday']]
y = df['attend']
```

In [33]:

```
#model building
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0
```

In [34]:

```
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

Out[34]:

```
LinearRegression()
```

In [35]:

```
#intercept
print('intercept: ',regressor.intercept_)
print('slope: ', regressor.coef_)
```

```
intercept:  -1.773063427617381e+18
slope:  [1.77306343e+18 1.77306343e+18 1.77306343e+18 1.77306343e+18
 1.77306343e+18 1.77306343e+18 1.77306343e+18]
```

In [36]:

```
#make predictions
y_pred = regressor.predict(X_test)
```

```
df_pred = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
df_pred
```

Out[36]:

	Actual	Predicted
22	44005	43264.0
27	51137	49408.0
61	39805	39680.0
13	32799	49408.0
71	43309	43520.0
74	35754	43264.0
30	50559	33536.0
55	32659	33536.0
53	46588	42240.0
26	38016	33536.0
50	52832	49408.0
42	53570	35328.0
48	39955	35328.0
33	40432	39680.0
73	42449	42240.0
2	28328	43520.0
57	37084	35328.0

In [37]:

```
from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
Mean Absolute Error: 5756.176470588235
Mean Squared Error: 74457521.47058824
Root Mean Squared Error: 8628.877184813111
```

In [45]:

```
"""The best day to run a marketing promotion is Tuesday because that is the day with th
```

Out[45]:

```
"The best day to run a marketing promotion is Tuesday because that is the day with the h
ighest average attendance, it is not even close to the other days. For a problem like th
is a model wouldn't even be ncessary because you know from the data that people choose t
o attend the dodger's games on Tuesdays. As we see here my model has a high Root Mean Sq
uared Error and this is because looking at the raw data set I can see very important fea
tures that will effect the model but for this situation I am not using them like if ther
e is a bobblehead or not for that game. My Linear Rgression model for the most part was 
accurate but I do know that if we were looking for more than just teh day of the week th
en more features could go into this model."
```

In [ ]: