In [1]:

```
import pandas as pd
import numpy as np
pd.options.display.float_format = '{:.2f}'.format
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)
#importing the data
df2022 = pd.read_csv("C://Users//12038//Desktop//ECON433//data//weekly_patterns_2022_s
df2021 = pd.read_csv("C://Users//12038//Desktop//ECON433//data//weekly_patterns_2021_s
df2020 = pd.read_csv("C://Users//12038//Desktop//ECON433//data//weekly_patterns_2020_s
df2019 = pd.read_csv("C://Users//12038//Desktop//ECON433//data//weekly_patterns_2019_s
df2018 = pd.read_csv("C://Users//12038//Desktop//ECON433//data//weekly_patterns_2018_s
#conditioning the data
df_new22 = df2022.loc[df2022['brands'] == "Domino's Pizza"]
df_new21 = df2021.loc[df2021['brands'] == "Domino's Pizza"]
df_new20 = df2020.loc[df2020['brands'] == "Domino's Pizza"]
df_new19 = df2019.loc[df2019['brands'] == "Domino's Pizza"]
df_new18 = df2018.loc[df2018['brands'] == "Domino's Pizza"]
dfdominos = pd.concat([df_new22, df_new21, df_new20, df_new19, df_new18],
                      ignore_index=True)
#splitting visits_by_day
dfdominos['visits_by_day'] = dfdominos['visits_by_day'].str.replace('[','')
dfdominos['visits_by_day'] = dfdominos['visits_by_day'].str.replace(']','')
dfdominos['dailyvisits_1'] = dfdominos['visits_by_day'].str.split(',').str[0]
dfdominos['dailyvisits_2'] = dfdominos['visits_by_day'].str.split(',').str[1]
dfdominos['dailyvisits_3'] = dfdominos['visits_by_day'].str.split(',').str[2]
dfdominos['dailyvisits_4'] = dfdominos['visits_by_day'].str.split(',').str[3]
dfdominos['dailyvisits_5'] = dfdominos['visits_by_day'].str.split(',').str[4]
dfdominos['dailyvisits_6'] = dfdominos['visits_by_day'].str.split(',').str[5]
dfdominos['dailyvisits_7'] = dfdominos['visits_by_day'].str.split(',').str[6]
#making dailyvisits an integer
dfdominos['dailyvisits_1'] = dfdominos['dailyvisits_1'].astype(int)
dfdominos['dailyvisits_2'] = dfdominos['dailyvisits_2'].astype(int)
dfdominos['dailyvisits_3'] = dfdominos['dailyvisits_3'].astype(int)
dfdominos['dailyvisits_4'] = dfdominos['dailyvisits_4'].astype(int)
dfdominos['dailyvisits_5'] = dfdominos['dailyvisits_5'].astype(int)
dfdominos['dailyvisits_6'] = dfdominos['dailyvisits_6'].astype(int)
dfdominos['dailyvisits_7'] = dfdominos['dailyvisits_7'].astype(int)
#reshaping the data and creation of dayofweek
dfdominos['id'] = dfdominos.index
dfdominos_long = pd.wide_to_long(dfdominos, ['dailyvisits'],
                               i = 'id' , j = 'dayofweek',
                               sep = '_')
#creation of outlier, manyvisits, and core_biz_area
dfdominos_long['outlier'] = np.where(
    dfdominos_long['dailyvisits'] >= 27.87, 1, 0)
dfdominos_long['manyvisits'] = np.where(
    dfdominos_long['dailyvisits'] >= 7, 1, 0)
dfdominos_long['core_biz_area'] = np.where(
    dfdominos_long['raw_visit_counts'] >= 200, 1, 0)
dfdominos_long.tail()
```

Out[1]:

		date_range_end	safegraph_brand_ids	raw_visit_counts	r
id	**dayofweek**				
299225	**7**	2018-05-07T00:00:00-04:00	SG_BRAND_da46ad6f82825669a56b44d32564dff8	70	
299226	**7**	2018-06-04T00:00:00-07:00	SG_BRAND_da46ad6f82825669a56b44d32564dff8	11	
299227	**7**	2018-05-07T00:00:00-05:00	SG_BRAND_da46ad6f82825669a56b44d32564dff8	2	
299228	**7**	2018-03-26T00:00:00-04:00	SG_BRAND_da46ad6f82825669a56b44d32564dff8	3	
299229	**7**	2018-09-17T00:00:00-06:00	SG_BRAND_da46ad6f82825669a56b44d32564dff8	9	

In [2]:

```
#converting to datetime and creating date and year variables
dfdominos_long = dfdominos_long.reset_index()
dfdominos_long['date_range_start'] = pd.to_datetime(dfdominos_long['date_range_start']
dfdominos_long['date'] = (dfdominos_long['date_range_start'] + pd.to_timedelta(dfdomir
dfdominos_long['date'] = dfdominos_long['date'].astype(str)
dfdominos_long['date'] = dfdominos_long['date'].str[:10]
dfdominos_long['year'] = dfdominos_long['date'].str[:4]
```

In [3]:

```
#creation of the weekend variable
dfdominos_long['weekend'] = np.where(
    dfdominos_long['dayofweek'].isin([6,7]), 1, 0)
```

In [4]:

```
#saving the long data to csv
dfdominos_long.to_csv('longdata.csv',
                 index = False)
```

In [5]:

```
#sanity check
df_long2 = dfdominos_long.loc[dfdominos_long['weekend'] == 1]
df_long3 = dfdominos_long.loc[dfdominos_long['weekend'] == 0]
print(len(dfdominos_long),len(df_long2),len(df_long3))
```

```
2094610 598460 1496150
```

In [6]:

```
#unique values of dayofweek
dfdominos_long['dayofweek'].nunique()
```

Out[6]:

```
7
```

Question 1

There are 7 unique values for day of week. There are 598,460 observations on the weekend and 1,496,150 observations on weekdays. These two values add to 2,094,610 , which is the length of the long dataframe.

In [7]:

```
#summary statistics of dailyvisits
dfdominos_long['dailyvisits'].describe()
```

Out[7]:

```
count   2094610.00
mean          5.43
std           7.48
min           0.00
25%           1.00
50%           4.00
75%           7.00
max        2813.00
Name: dailyvisits, dtype: float64
```

In [8]:

```
#determining the number of dailyvisits equal to zero, greater than 100 and 1000
zero_visit = dfdominos_long.loc[dfdominos_long['dailyvisits'] == 0]
hundred_visit = dfdominos_long.loc[dfdominos_long['dailyvisits'] >= 100]
thousand_visit = dfdominos_long.loc[dfdominos_long['dailyvisits'] >= 1000]
print(len(zero_visit),len(hundred_visit), len(thousand_visit))
```

```
269768 729 3
```

2 a.

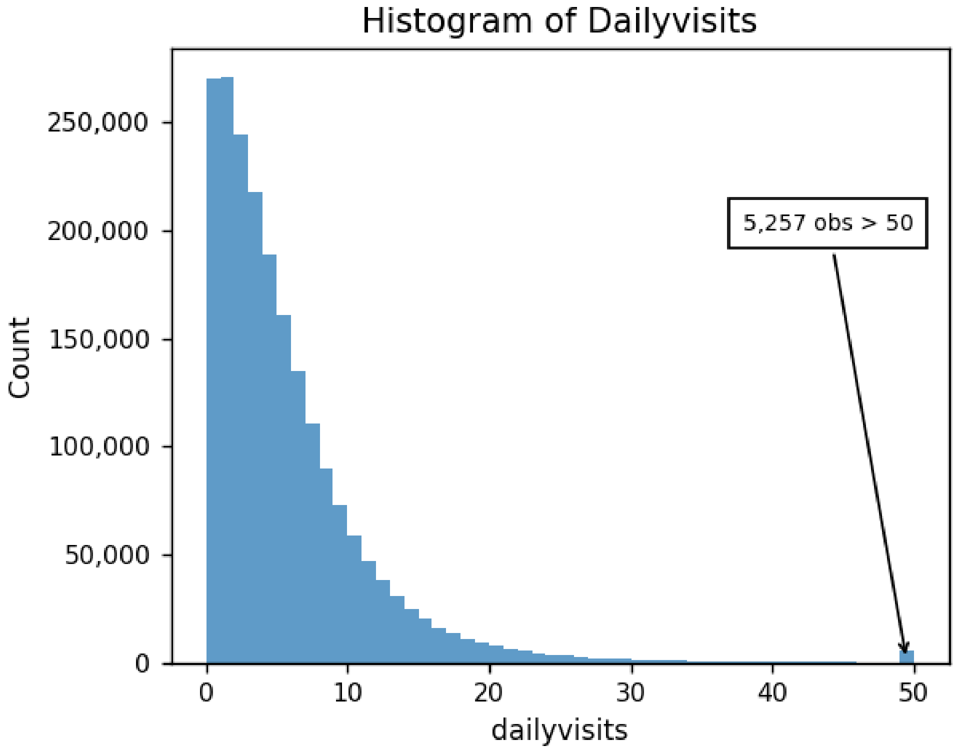
There are 2,094,610 observations of dailyvisits in the sample. The minimum is 0, the maximum is 2,813 , with an average of 5.43, and a median of 4. There are 269,768 observations equal to zero, 729 greater than or equal to 100, and 3 greater than or equal to 1000.

In [9]:

```
import matplotlib.pyplot as plt
%matplotlib notebook
```

In [10]:

```
#clipping values at 50
a = np.clip(dfdominos_long['dailyvisits'], 0, 50, out=None)
#plotting the histogram
plt.figure(figsize = (6, 4))
plt.hist(a, bins = 50, alpha = 0.7)
plt.subplots_adjust(left = 0.25)
plt.ylabel('Count')
plt.xlabel('dailyvisits')
plt.title('Histogram of Dailyvisits')
plt.yticks(fontsize = 9)
plt.xticks(fontsize = 9);
```



```
In [11]: #adding commas to the y axis
         import matplotlib.ticker as ticker
         plt.gca().yaxis.set_major_formatter(
             ticker.FuncFormatter(lambda x, p: format(int(x), ',')))
```

```
In [12]: #determining how many observations are greater than 50
         fifty_visit = dfdominos_long.loc[dfdominos_long['dailyvisits'] > 50]
         print(len(fifty_visit))
```

```
5257
```

```
In [13]: #plot annotation
         plt.annotate('5,257 obs > 50',
                      xy = (49.5, 1100),
                      xytext = (50, 200000),
                      arrowprops=dict(arrowstyle='->'),
                      bbox=dict(pad=5, fc = 'white'),
                      color = 'black', fontsize = 8,
                      horizontalalignment='right');
```

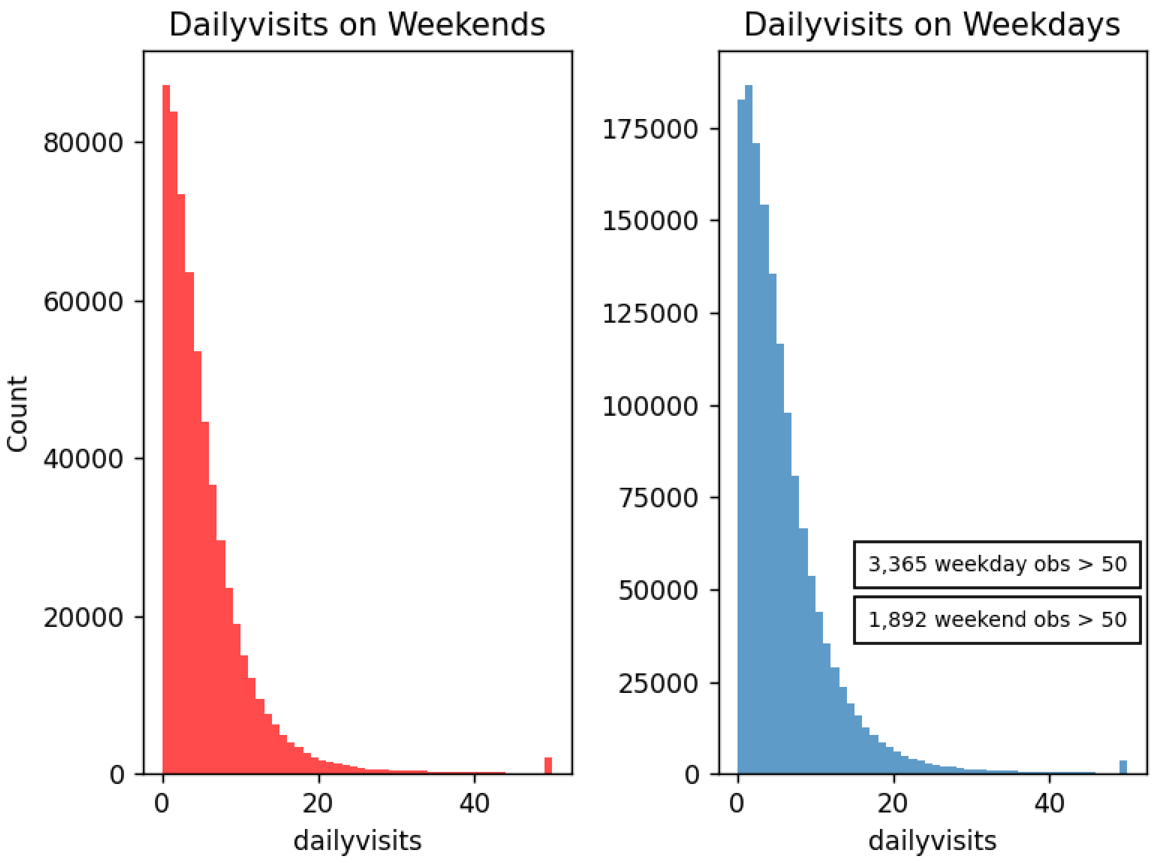
2 b.

I chose to focus on the subrange [0,50] because the data is very skewed and that seemed to be where the majority of my data is concentrated. From there I chose to use 50 bins such that each bar corresponded to an integer value of dailyvisits in my plot. I thought that using these definitions helped to make my graph as informative and readable as I could.

```
In [14]: #creating dataframes for weekends and weekdays
         long_weekend = dfdominos_long.loc[dfdominos_long['weekend'] == 1]
         long_weekday = dfdominos_long.loc[dfdominos_long['weekend'] == 0]
```

In [15]:

```
fig, (ax1, ax2) = plt.subplots(1,2)
#clipping values at 50
b = np.clip(long_weekend['dailyvisits'], 0, 50, out=None)
#creation of plot
ax1.hist(b,
         bins = 50,
         alpha = 0.7,
         range = [0,50],
         color = 'red');
ax1.set_title('Dailyvisits on Weekends');
ax1.set_ylabel('Count')
ax1.set_xlabel('dailyvisits')
#clipping values at 50
c = np.clip(long_weekday['dailyvisits'], 0, 50, out=None)
#creation of plot
ax2.hist(c,
         bins = 50,
         alpha = 0.7,
         range = [0,50]);
ax2.set_title('Dailyvisits on Weekdays');
ax2.set_xlabel('dailyvisits');
```



In [16]:

```
#determining the number of values greater than 50 for weekends and weekdays
weekend_fifty_visit = long_weekend.loc[long_weekend['dailyvisits'] > 50]
weekday_fifty_visit = long_weekday.loc[long_weekday['dailyvisits'] > 50]
print(len(weekend_fifty_visit), len(weekday_fifty_visit))
```

```
1892 3365
```

In [17]:

```
#formatting the figure and preventing the red warnings from showing up
fig.tight_layout()
```

In [18]:

```
#annotation of plots
plt.annotate('1,892 weekend obs > 50',
             xy = (49.5, 1100),
             xytext = (50, 40000),
             bbox=dict(pad=5, fc = 'white'),
             color = 'black', fontsize = 8,
             horizontalalignment='right');
plt.annotate('3,365 weekday obs > 50',
             xy = (49.5, 1100),
             xytext = (50, 55000),
             bbox=dict(pad=5, fc = 'white'),
             color = 'black', fontsize = 8,
             horizontalalignment='right');
```

2 c.

The distributions look largely similar, however it seems that the larger values for dailyvisits are more likely to occur on a weekend. Looking at the plot, the bar for 50 or more daily visits is larger. Despite there being less obervations on weekends, it appears that there is a greater proportion of observations greater than or equal to 50 as compared to the weekdays. Additionally, looking at the plot, the bars corresponding to the range of 35-45 dailyvisits on the weekends seem more pronounced when compared to weekdays, though this difference appears to be rather small.

In [19]:

```
#conversion to datetime
dfdominos_long['date'] = pd.to_datetime(dfdominos_long['date'])
#converting to day names
dfdominos_long['dayofweek'] = dfdominos_long['date'].dt.day_name()
#creating a table of total dailyvisits by day
agg0 = dfdominos_long.groupby(['dayofweek']).agg(
    {'dailyvisits': 'sum'})
agg0.head(7)
```

Out[19]:

	dailyvisits
dayofweek	
Friday	1979076
Monday	1512206
Saturday	1728225
Sunday	1312837
Thursday	1660068
Tuesday	1559139
Wednesday	1616561

3 a.

results shown above.

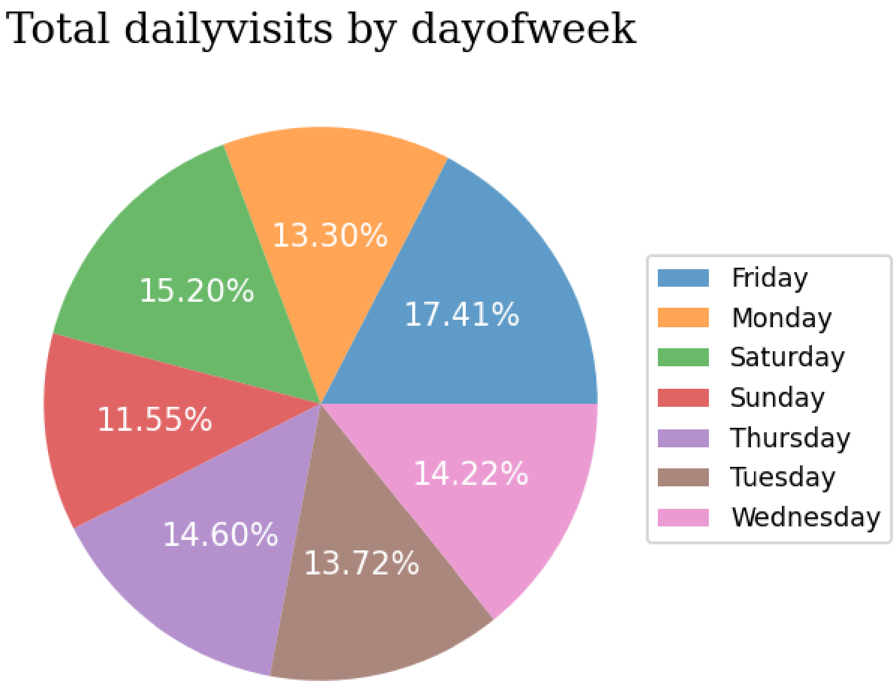
In [20]:

```
#creating a new dataframe containing the same information as above.
dv_sum = dfdominos_long[['dayofweek', 'dailyvisits']].groupby('dayofweek').sum()
```

In [21]:

```
#creation of the pie chart
plt.figure()

plt.pie(dv_sum['dailyvisits'],
        labels = dv_sum.index,
        autopct='%.2f%%',
        textprops={'color':'w', 'fontsize': 12},
        wedgeprops={'alpha':0.7})
plt.title('Total dailyvisits by dayofweek',
          fontsize = 16,
          fontname = 'serif');
```



In [22]:

```
#creation of legend
plt.legend(bbox_to_anchor=(1,0.5),
           loc="center right",
           fontsize=10,
           bbox_transform=plt.gcf().transFigure);
```

3 b.

Results shown above.

In [23]:
```
#creating a dataframe showing average dailyvisits by state
region_ = dfdominos_long[['dailyvisits', 'region']].groupby(['region']).mean()
region_.reset_index(inplace = True)
region_.head()
```

Out[23]:

	region	dailyvisits
0	AK	4.00
1	AL	9.19
2	AR	5.72
3	AZ	5.02
4	CA	4.80

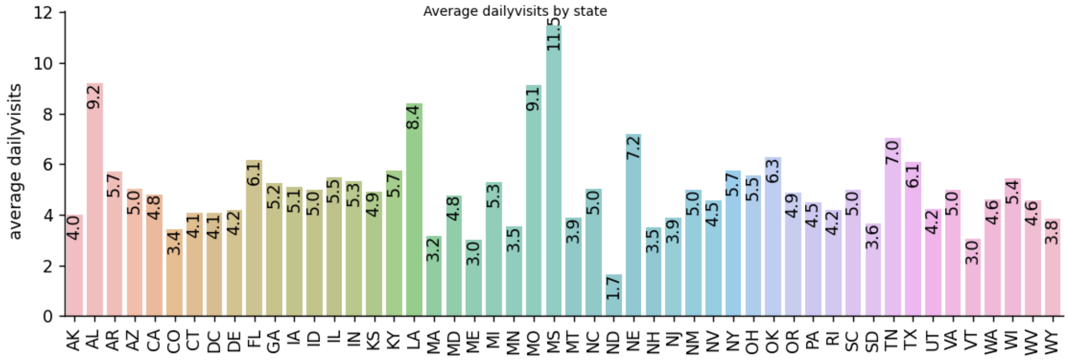
4 a.

results shown above

In [24]:
```
#creation of plot
import seaborn as sns

myplot = sns.catplot(
    data=region_, kind='bar',
    x='region', y='dailyvisits',
    order = ['AK', 'AL', 'AR', 'AZ', 'CA', 'CO', 'CT', 'DC',
             'DE', 'FL', 'GA', 'IA', 'ID', 'IL', 'IN', 'KS', 'KY',
             'LA', 'MA', 'MD', 'ME', 'MI', 'MN', 'MO', 'MS', 'MT',
             'NC', 'ND', 'NE', 'NH', 'NJ', 'NM', 'NV', 'NY', 'OH',
             'OK', 'OR', 'PA', 'RI', 'SC', 'SD', 'TN', 'TX', 'UT',
             'VA', 'VT', 'WA', 'WI', 'WV', 'WY'],
    alpha=.6)

myplot.set_axis_labels("", " average dailyvisits")
myplot.fig.suptitle("Average dailyvisits by state",
                    fontsize=8);
plt.gcf().set_size_inches(10,3)
plt.xticks(rotation = 90);
```



```
C:\Users\12038\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The
figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

In [25]:

```
#adding the values on top of the bars
myplot.axes[0,0].bar_label(myplot.axes[0,0].containers[0],
                           fmt = '%.1f',
                           padding = -15,
                           color = 'black',
                          rotation = 90);
```

4 b.

results shown above

In [26]:

```
#creation of postcovid variable
dfdominos_long['postcovid'] = np.where(
    dfdominos_long['date'] > '2020-03-13', 1, 0)
#creating a dataframe showing average dailyvisits by state pre and post covid
r_w = (dfdominos_long[['dailyvisits', 'region', 'postcovid']]
                  .groupby(['region', 'postcovid']).mean())
r_w.reset_index(inplace = True)
r_w.head()
```

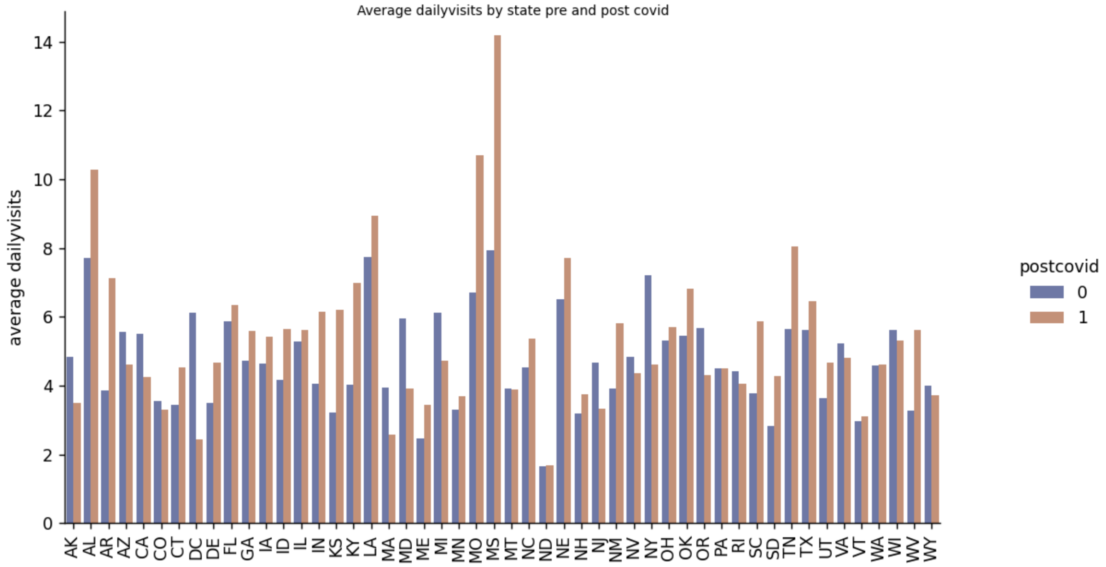
Out[26]:

	region	postcovid	dailyvisits
0	AK	0	4.83
1	AK	1	3.50
2	AL	0	7.71
3	AL	1	10.29
4	AR	0	3.86

In [49]:

```
#creation of the plot
ax = sns.catplot(
    data = r_w, kind="bar",
    x="region", y="dailyvisits",
    hue="postcovid",
    palette = 'dark', alpha=.6,
    order = ['AK', 'AL', 'AR', 'AZ', 'CA', 'CO', 'CT', 'DC',
             'DE', 'FL', 'GA', 'IA', 'ID', 'IL', 'IN', 'KS', 'KY',
             'LA', 'MA', 'MD', 'ME', 'MI', 'MN', 'MO', 'MS', 'MT',
             'NC', 'ND', 'NE', 'NH', 'NJ', 'NM', 'NV', 'NY', 'OH',
             'OK', 'OR', 'PA', 'RI', 'SC', 'SD', 'TN', 'TX', 'UT',
             'VA', 'VT', 'WA', 'WI', 'WV', 'WY'])

ax.set_axis_labels("", "average dailyvisits")
ax.fig.suptitle("Average dailyvisits by state pre and post covid ",
                fontsize=8);
plt.gcf().set_size_inches(10,5)
plt.xticks(rotation = 90);
```



```
C:\Users\12038\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The
figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

4 c.

This bar chart is interesting because it shows that covid actually seemed to boost business in some areas. Some differences can be seen in the different geographic areas of the United States. Some southern states like Alabama, Florida, and Missouri actually saw a large uptick in business. In contrast, some northern states like Massachusetts, Maine, and New Jersey saw a downturn in business. Other states and areas such Washington and Wisconsin seemed more resistant to deviations in business due to covid. I imagine these trends likely have to do with how strict lockdown and covid response was in the particular state, as well as if they were Democrat or Republican run states.

In [28]:

```
#creating year-month variable
dfdominos_long['year-month'] = pd.to_datetime(dfdominos_long['date'].astype(str).str[:
#creation of dataframe containing mean monthly visits for every month in the sample
d1 = dfdominos_long.groupby(['placekey', 'year-month'])['dailyvisits'].sum().reset_inc
monthly_visits = d1.groupby('year-month')['dailyvisits'].mean().reset_index().rename(c
monthly_visits.head()
```

Out[28]:

	year-month	monthly_visits
0	2018-01-01	88.06
1	2018-02-01	94.26
2	2018-03-01	121.88
3	2018-04-01	142.63
4	2018-05-01	142.08

5 a.

results shown above

In [29]:

```
import matplotlib.dates as mdates
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m-%d"))
```

In [30]:

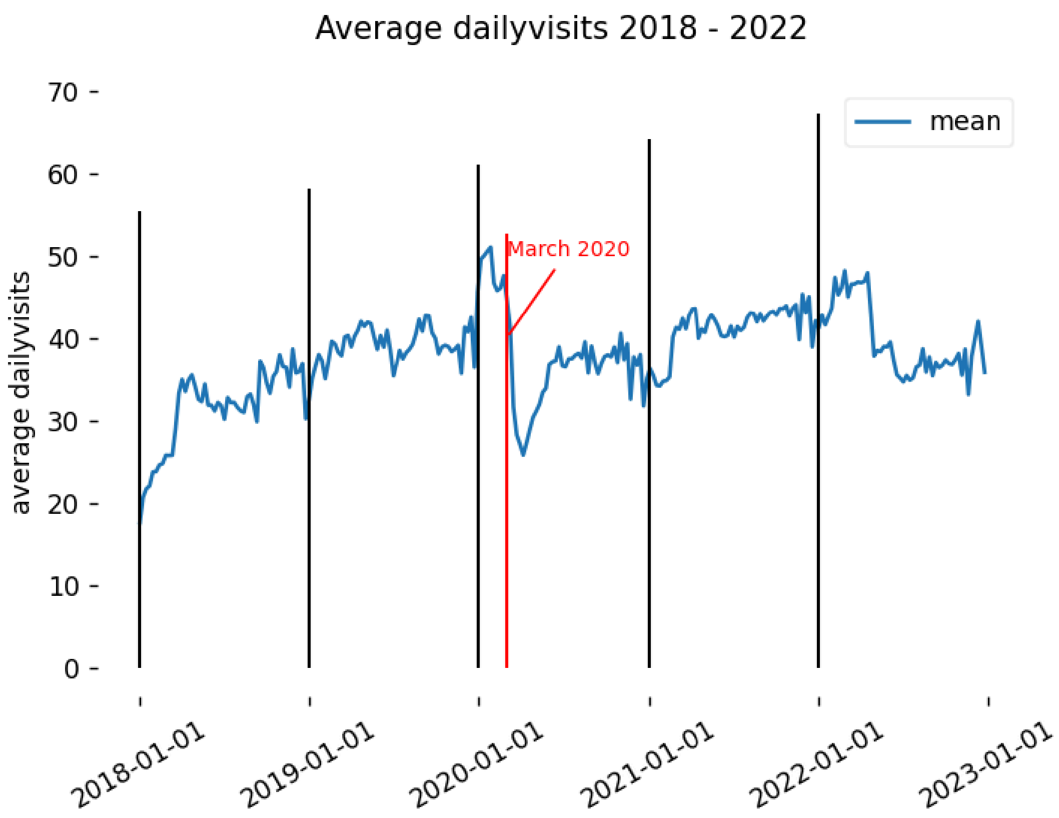
```
#cleaning up date_range_start
dfdominos_long['date_range_start'] = pd.to_datetime(dfdominos_long['date_range_start']
#creation of dataframe showing average visits by week
weekly_visits0 = (dfdominos_long.groupby('date_range_start').agg(
    {'raw_visit_counts': 'mean'}))
weekly_visits0.head()
```

Out[30]:

	raw_visit_counts
date_range_start	
2018-01-01	17.69
2018-01-08	20.77
2018-01-15	21.80
2018-01-22	22.14
2018-01-29	23.83

In [31]:

```
#creation of the plot
plt.figure()
plt.plot(weekly_visits0['raw_visit_counts'], '-', label = 'mean')
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m-%d"))
plt.xticks(rotation=30)
plt.subplots_adjust(bottom = 0.2)
plt.legend(loc = 0, facecolor='white',
           framealpha = 0.3)
plt.ylabel('average dailyvisits')
plt.title('Average dailyvisits 2018 - 2022', y=1.05);
```



In [32]:
```
#partition of plot by year with a line denoting March 2020
plt.vlines(x = pd.to_datetime('2020-03-01'),
           ymin = 0, ymax = plt.ylim()[1],
          linestyle = '-', color = 'r', lw = 1.2)
plt.vlines(x = pd.to_datetime('2018-01-01'),
           ymin = 0, ymax = plt.ylim()[1],
          linestyle = '-', color = 'black', lw = 1.2)
plt.vlines(x = pd.to_datetime('2019-01-01'),
           ymin = 0, ymax = plt.ylim()[1],
          linestyle = '-', color = 'black', lw = 1.2)
plt.vlines(x = pd.to_datetime('2020-01-01'),
           ymin = 0, ymax = plt.ylim()[1],
          linestyle = '-', color = 'black', lw = 1.2)
plt.vlines(x = pd.to_datetime('2021-01-01'),
           ymin = 0, ymax = plt.ylim()[1],
          linestyle = '-', color = 'black', lw = 1.2)
plt.vlines(x = pd.to_datetime('2022-01-01'),
           ymin = 0, ymax = plt.ylim()[1],
          linestyle = '-', color = 'black', lw = 1.2);
```

In [33]:
```
#annotation of plot
plt.annotate('March 2020',
             xy = (pd.to_datetime('2020-03-01'), 40),
             xytext= (pd.to_datetime('2020-03-01'), 50),
             arrowprops=dict(arrowstyle = '-', color = 'red'),
            color = 'red', fontsize = 8);
```

In [34]:
```
#making the plot look nicer
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
plt.gca().spines['left'].set_visible(False)
plt.gca().spines['bottom'].set_visible(False)
```

5 b.

results shown above

In [35]:

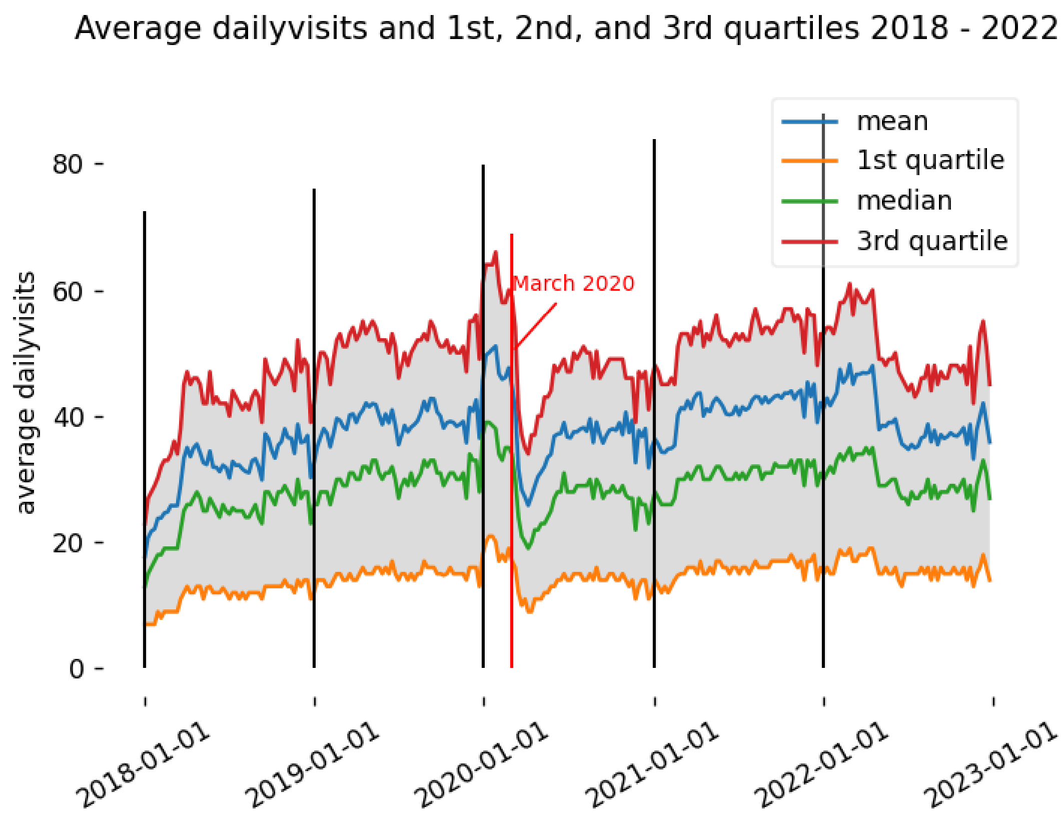
```
#defining the quartiles
def q25(x):
    return x.quantile(0.25)
def q50(x):
    return x.quantile(0.5)
def q75(x):
    return x.quantile(0.75)
#creation of dataframe showing quartiles and mean of dailyvisits by week
weekly_visits = (dfdominos_long.groupby('date_range_start').agg(
    {'raw_visit_counts': [q25, q50, q75, 'mean']}))
weekly_visits.columns = [f"{x}_{y}" for x, y in weekly_visits.columns.to_flat_index()]
weekly_visits.head()
```

Out[35]:

	raw_visit_counts_q25	raw_visit_counts_q50	raw_visit_counts_q75	raw_visit_counts_mea
date_range_start				
2018-01-01	7.00	13.00	23.00	17.6
2018-01-08	7.00	15.00	27.00	20.7
2018-01-15	7.00	16.00	28.00	21.8
2018-01-22	7.00	17.00	29.00	22.1
2018-01-29	9.00	18.00	30.25	23.8

In [36]:

```
#plotting the figure
plt.figure()
plt.plot(weekly_visits['raw_visit_counts_mean'], '-', label = 'mean')
plt.plot(weekly_visits['raw_visit_counts_q25'], '-', label = '1st quartile')
plt.plot(weekly_visits['raw_visit_counts_q50'], '-', label = 'median')
plt.plot(weekly_visits['raw_visit_counts_q75'], '-', label = '3rd quartile')
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m-%d"))
plt.xticks(rotation=30)
plt.subplots_adjust(bottom = 0.2)
plt.legend(loc = 0, facecolor='white',
           framealpha = 0.3)
plt.ylabel('average dailyvisits')
plt.title('Average dailyvisits and 1st, 2nd, and 3rd quartiles 2018 - 2022', y=1.05);
```



Average dailyvisits and 1st, 2nd, and 3rd quartiles 2018 - 2022

In [37]:

```
# partition of plot by year with a line denoting March 2020
plt.vlines(x = pd.to_datetime('2020-03-01'),
           ymin = 0, ymax = plt.ylim()[1],
          linestyle = '-', color = 'r', lw = 1.2)
plt.vlines(x = pd.to_datetime('2018-01-01'),
           ymin = 0, ymax = plt.ylim()[1],
          linestyle = '-', color = 'black', lw = 1.2)
plt.vlines(x = pd.to_datetime('2019-01-01'),
           ymin = 0, ymax = plt.ylim()[1],
          linestyle = '-', color = 'black', lw = 1.2)
plt.vlines(x = pd.to_datetime('2020-01-01'),
           ymin = 0, ymax = plt.ylim()[1],
          linestyle = '-', color = 'black', lw = 1.2)
plt.vlines(x = pd.to_datetime('2021-01-01'),
           ymin = 0, ymax = plt.ylim()[1],
          linestyle = '-', color = 'black', lw = 1.2)
plt.vlines(x = pd.to_datetime('2022-01-01'),
           ymin = 0, ymax = plt.ylim()[1],
          linestyle = '-', color = 'black', lw = 1.2);
```

In [38]:

```
#filling the area between 1st and 3rd quartiles
plt.gca().fill_between(weekly_visits.index,
                       weekly_visits['raw_visit_counts_q25'],
                       weekly_visits['raw_visit_counts_q75'],
                       facecolor = 'grey', alpha = 0.25);
```

In [39]:

```
#annotation of plot
plt.annotate('March 2020',
             xy = (pd.to_datetime('2020-03-01'), 50),
             xytext= (pd.to_datetime('2020-03-01'), 60),
             arrowprops=dict(arrowstyle = '-', color = 'red'),
            color = 'red', fontsize = 8);
```

In [40]:
```
#making the plot look nicer
plt.gca().spines['right'].set_visible(False)
plt.gca().spines['top'].set_visible(False)
plt.gca().spines['left'].set_visible(False)
plt.gca().spines['bottom'].set_visible(False)
```

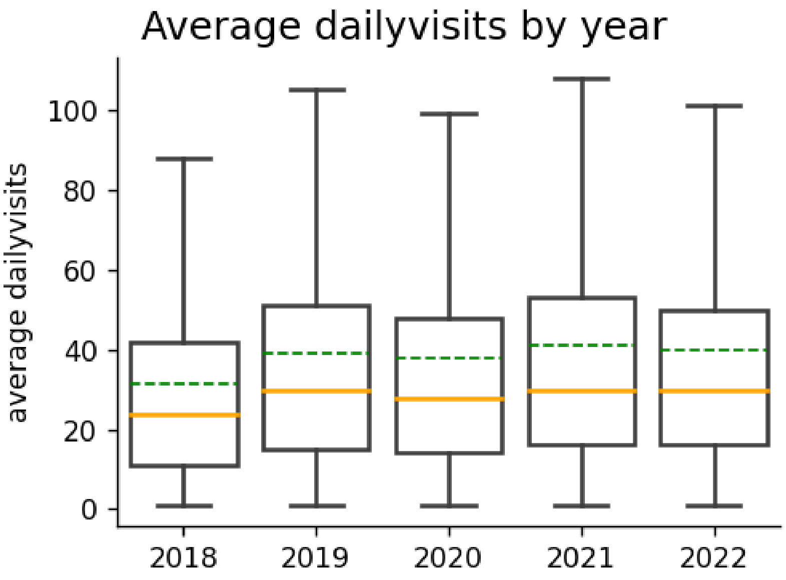
5 c.

One trend from the line graph is the sharp decline in dailyvisits after March 2020 followed by a recovery to pre covid levels through to 2022. Sometime in or around the first quarter of 2022, business took another downturn. Another observation from the line graph is that at some time around the first quarter of 2018, average dailyvisits rose quickly and dramatically.

In [41]:
```
#creation of the boxplot for each year
PROPS = {
    'boxprops':{'facecolor':'none'},
    'medianprops':{'color':'orange'},
    'meanprops': {'color':'green', 'ls': '--'},
    'flierprops': {'marker': 'o'}
}

mybox = sns.catplot(
    data=dfdominos_long, kind='box',
    x='year', y='raw_visit_counts',
    order = ['2018', '2019', '2020', '2021', '2022'],
    showmeans=True,
    meanline=True,
    showfliers=False,
    **PROPS)

mybox.set_axis_labels("", "average dailyvisits")
mybox.fig.suptitle("Average dailyvisits by year",
                   fontsize=14);
plt.gcf().set_size_inches(4,3)
plt.subplots_adjust(top = 0.9)
```



```
C:\Users\12038\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The
figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

6.

Taking a look at this chart, 2021 seemed to be the highest perfoming year in terms of dailyvisits. It has the highest mean, median, 1st and 3rd quartiles, and maximum value. In comparison to the line graph from the last question, this seems to check out. Looking at all the lines within the time period of 2021 on the graph, they do all seem to be slightly higher on average in comparison to other years on the line graph. One difference between these plots is that the line graph shows the sharp decline in dailyvisits due to covid much better than the boxplot does due the continuous nature of the x axis in the line graph.

In [42]:

```
pip install geopandas -q
```

```
Note: you may need to restart the kernel to use updated packages.
```

In [43]:

```
pip install geoplot -q
```

```
Note: you may need to restart the kernel to use updated packages.
```

In [44]:

```
#getting rid of the red warning
import warnings
warnings.filterwarnings("ignore", category=FutureWarning)
#importing geographic data and creating dataframe
import geopandas as gpd
import geoplot as gplt
geoData = gpd.read_file('https://raw.githubusercontent.com/holtzy/The-Python-Graph-Gal
geoData.id = geoData.id.astype(str).astype(int)
stateToRemove = ['02', '15', '72']
geoData = geoData[~geoData.STATE.isin(stateToRemove)]
geoData = geoData.explode()
```

In [45]:

```
#creating an id number for each state
state_id = {
    'AL': '01', 'AK': '02', 'AZ': '04', 'AR': '05', 'CA': '06',
    'CO': '08', 'CT': '09', 'DE': '10', 'FL': '12', 'GA': '13',
    'HI': '15', 'ID': '16', 'IL': '17', 'IN': '18', 'IA': '19',
    'KS': '20', 'KY': '21', 'LA': '22', 'ME': '23', 'MD': '24',
    'MA': '25', 'MI': '26', 'MN': '27', 'MS': '28', 'MO': '29',
    'MT': '30', 'NE': '31', 'NV': '32', 'NH': '33', 'NJ': '34',
    'NM': '35', 'NY': '36', 'NC': '37', 'ND': '38', 'OH': '39',
    'OK': '40', 'OR': '41', 'PA': '42', 'RI': '44', 'SC': '45',
    'SD': '46', 'TN': '47', 'TX': '48', 'UT': '49', 'VT': '50',
    'VA': '51', 'WA': '53', 'WV': '54', 'WI': '55', 'WY': '56',
}

#creating a state_code variable for the long data corresponding to the state
dfdominos_long['state_code'] = dfdominos_long['region'].map(state_id)
```

In [46]:

```
#making the long data smaller to prevent memory issues
df_long_summary = dfdominos_long.groupby(['state_code']).agg(
    {'dailyvisits': 'mean'})
df_long_summary = df_long_summary.reset_index()
df_long_summary.head()
```

Out[46]:

	state_code	dailyvisits
0	01	9.19
1	02	4.00
2	04	5.02
3	05	5.72
4	06	4.80

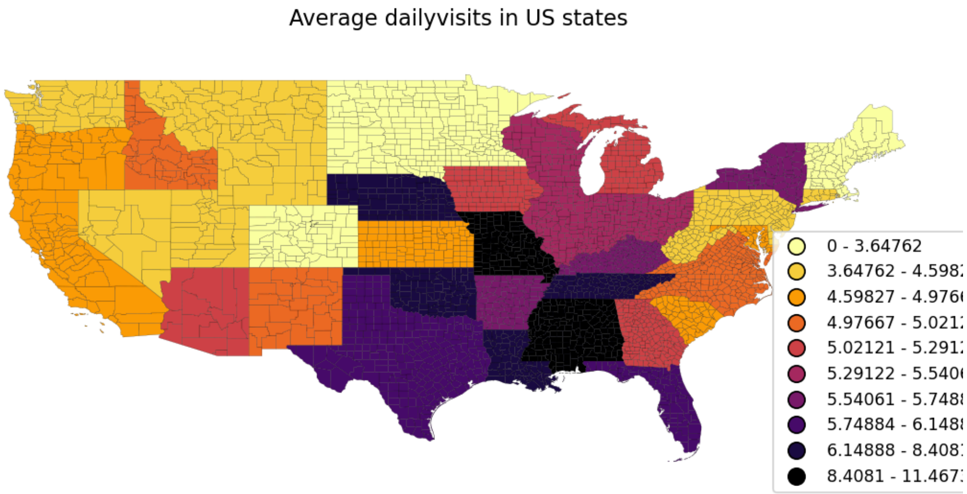
In [47]:

```
#merging the geographic data with the long data
fullData = pd.merge(geoData, df_long_summary, left_on=['STATE'], right_on=['state_code
```

In [75]:

```
#creating the map
fig, ax = plt.subplots(1, 1, figsize=(10, 8))
import mapclassify as mc
scheme = mc.Quantiles(fullData['dailyvisits'], k=10)
gplt.choropleth(fullData,
    hue="dailyvisits",
    linewidth=.1,
    scheme=scheme, cmap='inferno_r',
    legend=True,
    edgecolor='black',
    ax=ax
);

ax.set_title(' Average dailyvisits in US states', fontsize=13);
#moving the legend so that it won't cover the map
plt.subplots_adjust(right=1.027)
```


Average dailyvisits in US states
● 0 - 3.64762
● 3.64762 - 4.598
● 4.59827 - 4.976
● 4.97667 - 5.021
● 5.02121 - 5.291
● 5.29122 - 5.540
● 5.54061 - 5.748
● 5.74884 - 6.148
● 6.14888 - 8.408
● 8.4081 - 11.467

7.

a)I would say this chart passes the smell test based on public information. As we cans see on the map, southern states seem to have the heaviest traffic in terms of dailyvisits. This makes sense according to data found online. Texas has the most locations and other southern states also top the list in terms of number of locations, so it would make sense for Domino's to be popular among states in this geographic area. b) A lot can be garnered from this graphic. First, it appears that Domino's is busier in the eastern part of the country. In general, the colors on the right half of the map are darker than on the left side, indicating higher average dailyvisits overall for the eastern part of the country. Additionally, this same trend seems to be true when speaking on the northern half, vs southern half of the country. In general, the southern part of the country displays a darker color for any given state, indicating higher average dailyvisits. c) This information can be useful in deciding state to state strategies for Domino's. For instance, you can take a look at some of the southern states that you notice have higher average dailyvisits, analyze their business practices, and try to implement those strategies in northern states in order to boost their numbers. Alternatively, a state like New York has higher than average dailyvisits when compared to states in the surrounding area. What makes New York different than other northern states? How can we use what we learn about analyzing the practices of restaurants in New York and apply them to the surrounding states to boost business? Looking at the data represented geographically allows decision makers to ask questions and do analysis to be more informed when deciding strategies state to state.