0.0.1 Question 0

Question 0A What is the granularity of the data (i.e. what does each row represent)?

Each row represents the number of bike riders observed during a specific hour of a day. We track different data points such as temperature, season, etc for each hour as well as collecting time data for each record.

Question 0B For this assignment, we'll be using this data to study bike usage in Washington D.C. Based on the granularity and the variables present in the data, what might some limitations of using this data be? What are two additional data categories/variables that you can collect to address some of these limitations?

A potential issue regarding the data is we are not considering duplicate counts during the hour, for example if the same casual rider has to rent a bike twice in one hour. Due to this we may not have accurate data regarding total number of bikers observed. In order to combat this we can include further information about riders to have more granularity, such as renter name/ renter data.

Another thing to consider is to have multiple records representing rentals from different areas. Having data like this can better explain the effect of weather when more data points are kept constant. Thus including more data from different locations, and adding a location category to data can be helpful.

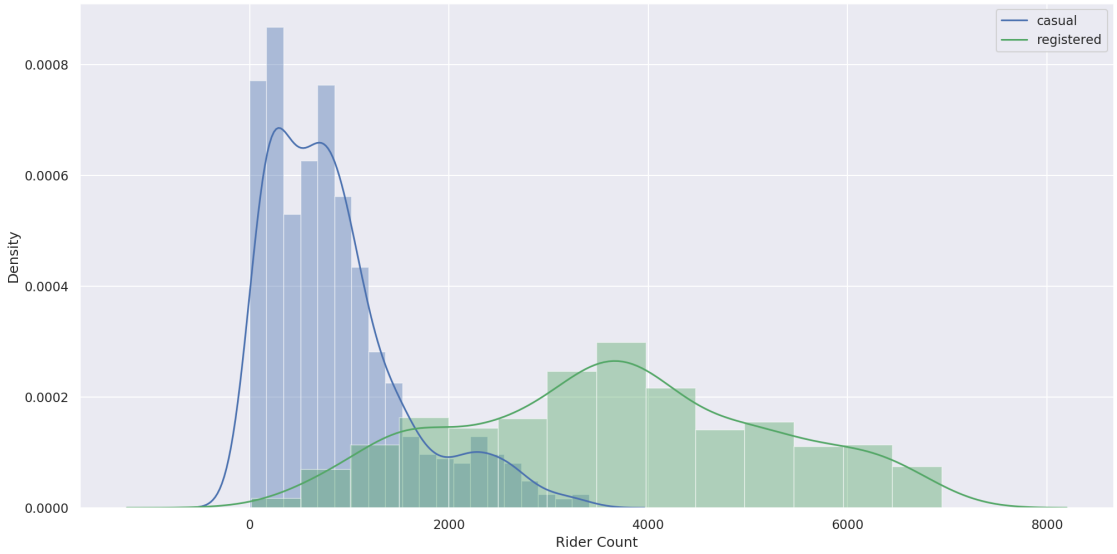
0.0.2 Question 2

Question 2a Use the `sns.distplot` function to create a plot that overlays the distribution of the daily counts of bike users, using blue to represent `casual` riders, and green to represent `registered` riders. The temporal granularity of the records should be daily counts, which you should have after completing question 1c. **You can ignore all warnings that say `distplot` is a deprecated function.**

Include a legend, xlabel, ylabel, and title. Read the seaborn plotting tutorial if you're not sure how to add these. After creating the plot, look at it and make sure you understand what the plot is actually telling us, e.g on a given day, the most likely number of registered riders we expect is ~4000, but it could be anywhere from nearly 0 to 7000.

```
In [41]: sns.distplot(daily_counts['casual'], color = 'b')
         sns.distplot(daily_counts['registered'], color = 'g')
         plt.xlabel('Rider Count')
         plt.ylabel('Density')
         plt.legend(['casual', 'registered'])
         plt.show()
```

```
/opt/conda/lib/python3.8/site-packages/seaborn/distributions.py:2551: FutureWarning: `distplot` is a dep
  warnings.warn(msg, FutureWarning)
/opt/conda/lib/python3.8/site-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support for mul
  ndim = x[:, None].ndim
/opt/conda/lib/python3.8/site-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for multi-di
  x = x[:, np.newaxis]
/opt/conda/lib/python3.8/site-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for multi-di
  y = y[:, np.newaxis]
/opt/conda/lib/python3.8/site-packages/seaborn/distributions.py:2551: FutureWarning: `distplot` is a dep
  warnings.warn(msg, FutureWarning)
/opt/conda/lib/python3.8/site-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support for mul
  ndim = x[:, None].ndim
/opt/conda/lib/python3.8/site-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for multi-di
  x = x[:, np.newaxis]
/opt/conda/lib/python3.8/site-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for multi-di
  y = y[:, np.newaxis]
```


casual
registered
Density
0.0008
0.0006
0.0004
0.0002
0.0000
0
2000
4000
6000
8000
Rider Count

0.0.3 Question 2b

In the cell below, descibe the differences you notice between the density curves for casual and registered riders. Consider concepts such as modes, symmetry, skewness, tails, gaps and outliers. Include a comment on the spread of the distributions.

Registered riders count is a symmetrical distribution with no visual skew. It is unimodat at around 3800-3900 riders and does not have outliers and gaps. The graph has a large spread from 0 to 7000 riders.

Unregistered riders is fairly asymmetrical with two modes at 200 and 500. It has a large skew to the right but does not contain gaps or outliers but has a fairly small spread from 0 to 4000 riders.

0.0.4 Question 2c

The density plots do not show us how the counts for registered and casual riders vary together. Use `sns.lmplot` to make a scatter plot to investigate the relationship between casual and registered counts. This time, let's use the `bike` DataFrame to plot hourly counts instead of daily counts.

The `lmplot` function will also try to draw a linear regression line (just as you saw in Data 8). Color the points in the scatterplot according to whether or not the day is a working day (your colors do not have to match ours exactly, but they should be different based on whether the day is a working day).

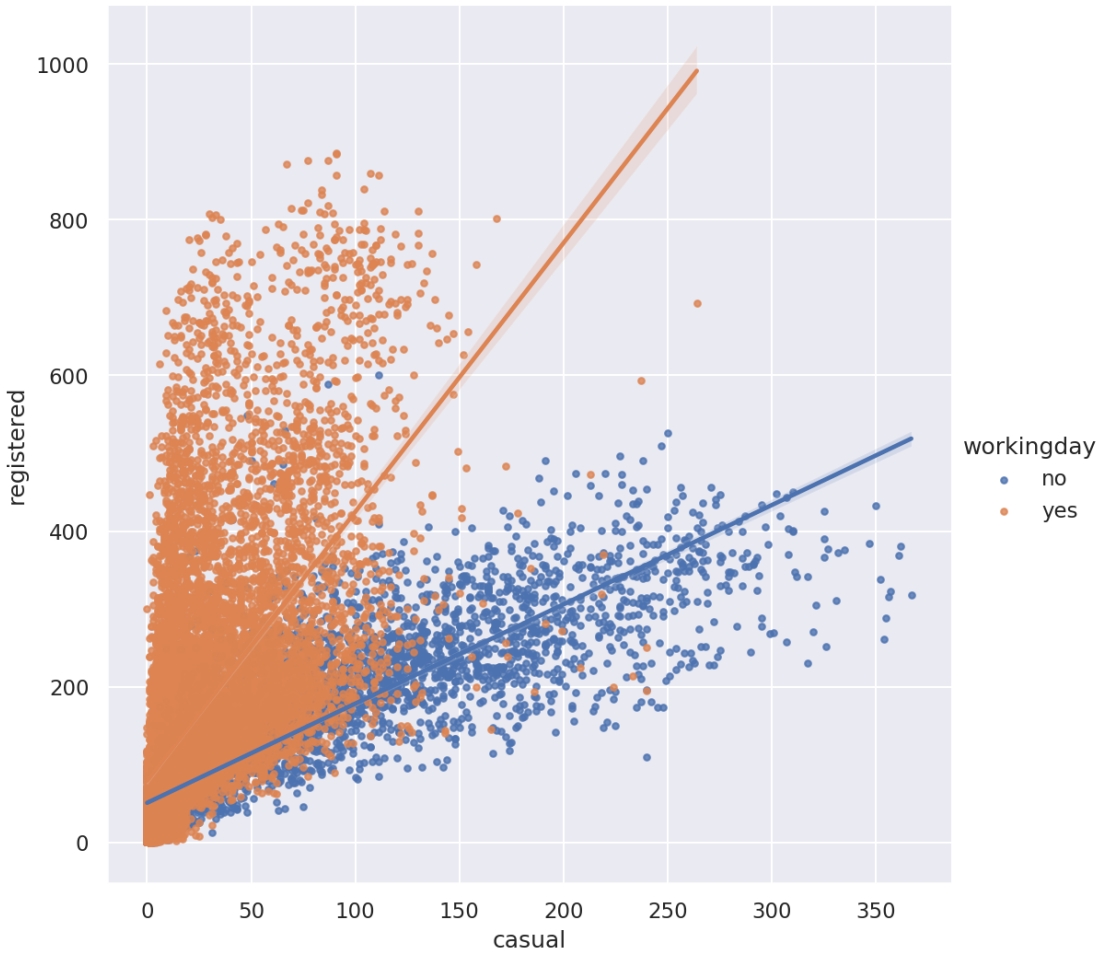
There are many points in the scatter plot, so make them small to help reduce overplotting. Also make sure to set `fit_reg=True` to generate the linear regression line. You can set the `height` parameter if you want to adjust the size of the `lmplot`.

Hints: * Checkout this helpful tutorial on `lmplot`.

- You will need to set `x`, `y`, and `hue` and the `scatter_kws`.

In [44]:
```
# Make the font size a bit bigger
sns.set(font_scale=1)
sns.lmplot(x = 'casual', y = 'registered', data = bike, size = 7, hue='workingday', scatter_kws

plt.show()
```

```
/opt/conda/lib/python3.8/site-packages/seaborn/regression.py:580: UserWarning: The `size` parameter has
  warnings.warn(msg, UserWarning)
```


registered
casual
workingday
no
yes
0
200
400
600
800
1000
0
50
100
150
200
250
300
350

0.0.5 Question 2d

What does this scatterplot seem to reveal about the relationship (if any) between casual and registered riders and whether or not the day is on the weekend? What effect does overplotting have on your ability to describe this relationship?

Based on the slopes of the line of best fits, there seems to be a correlation between people renting bikes on working dats and non working days. On workingdays, a lot more registered bikers come out compared to casual bikers while on weekends there seems to be more casual bikers.

Overplotting makes it difficult to see the points from 0 to 100 as there is a lot of points situated there making it hard to visualize early correlation of blue points.

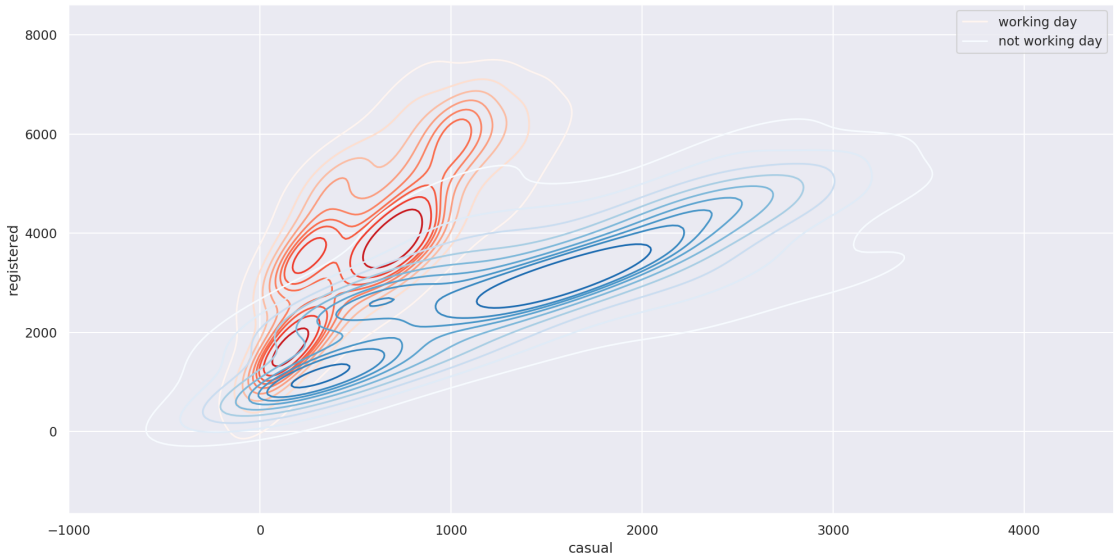
Generating the plot with weekend and weekday separated can be complicated so we will provide a walk-through below, feel free to use whatever method you wish if you do not want to follow the walkthrough.

Hints: * You can use `loc` with a boolean array and column names at the same time * You will need to call kdeplot twice. * Check out this guide to see an example of how to create a legend. In particular, look at how the example in the guide makes use of the `label` argument in the call to `plt.plot()` and what the `plt.legend()` call does. This is a good exercise to learn how to use examples to get the look you want. * You will want to set the `cmap` parameter of `kdeplot` to `"Reds"` and `"Blues"` (or whatever two contrasting colors you'd like). You are required for this question to use two sets of contrasting colors for your plots.

After you get your plot working, experiment by setting `shade=True` in `kdeplot` to see the difference between the shaded and unshaded version. Please submit your work with `shade=False`.

In [68]:
```
# Set 'is_workingday' to a boolean array that is true for all working_days
is_workingday = bike[bike['workingday'] == 'yes']

# Bivariate KDEs require two data inputs.
# In this case, we will need the daily counts for casual and registered riders on workdays
casual_workday = is_workingday.groupby("dteday").agg(sum)['casual']
registered_workday =  is_workingday.groupby("dteday").agg(sum)['registered']

# Use sns.kdeplot on the two variables above to plot the bivariate KDE for weekday rides
sns.kdeplot(casual_workday, registered_workday, cmap = 'Reds', label = "working day")

# Repeat the same steps above but for rows corresponding to non-workingdays
is_nonworkingday = bike[bike['workingday'] == 'no']
casual_non_workday = is_nonworkingday.groupby("dteday").agg(sum)['casual']
registered_non_workday = is_nonworkingday.groupby("dteday").agg(sum)['registered']

# Use sns.kdeplot on the two variables above to plot the bivariate KDE for non-workingday ride
sns.kdeplot(casual_non_workday,registered_non_workday, cmap = 'Blues',label = "not working day
plt.legend()
plt.show()
```

```
/opt/conda/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following vari
  warnings.warn(
/opt/conda/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following vari
  warnings.warn(
```


working day
not working day
8000
6000
4000
2000
0
registered
−1000
0
1000
2000
3000
4000
casual

Question 3b What additional details can you identify from this contour plot that were difficult to determine from the scatter plot?

By using the density visualization graph, we can see a much clearer linear association between casual and registered bike renters. We can also better visualize the overlap near (0,0) and better see which distribution the points belong too.

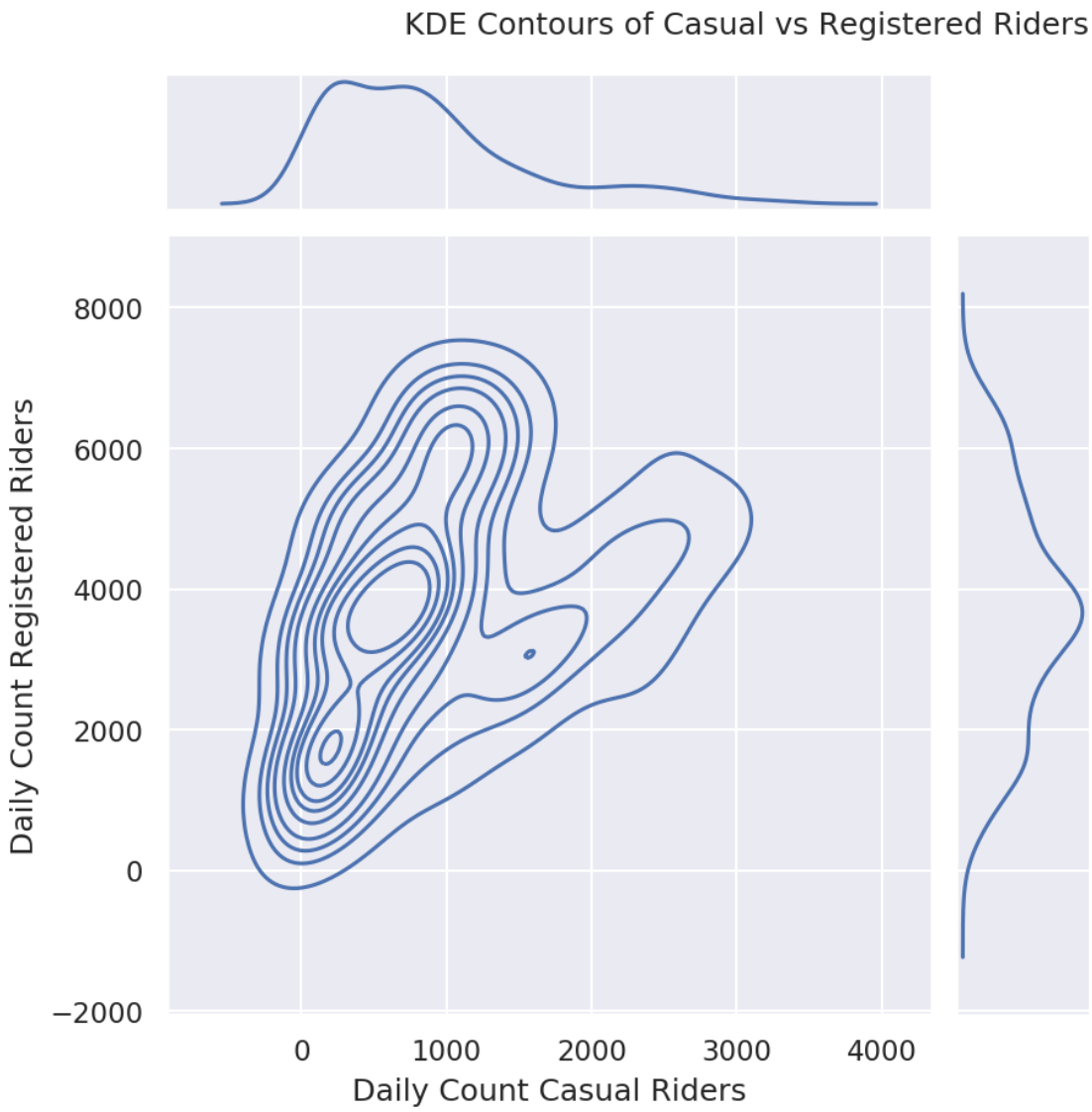
0.1 4: Joint Plot

As an alternative approach to visualizing the data, construct the following set of three plots where the main plot shows the contours of the kernel density estimate of daily counts for registered and casual riders plotted together, and the two "margin" plots (at the top and right of the figure) provide the univariate kernel density estimate of each of these variables. Note that this plot makes it harder see the linear relationships between casual and registered for the two different conditions (weekday vs. weekend).

Hints: * The seaborn plotting tutorial has examples that may be helpful. * Take a look at `sns.jointplot` and its `kind` parameter. * `set_axis_labels` can be used to rename axes on the contour plot. * `plt.suptitle` from lab 1 can be handy for setting the title where you want. * `plt.subplots_adjust(top=0.9)` can help if your title overlaps with your plot

In [69]:
```
sns.jointplot(x = 'casual', y = 'registered', data = daily_counts, kind = 'kde').set_axis_labe


        plt.title('KDE Contours of Casual vs Registered Riders', loc = 'right' , pad = 80)
        plt.show()
```

```
/opt/conda/lib/python3.8/site-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support for mu
  ndim = x[:, None].ndim
/opt/conda/lib/python3.8/site-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for multi-d
  x = x[:, np.newaxis]
/opt/conda/lib/python3.8/site-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for multi-d
  y = y[:, np.newaxis]
/opt/conda/lib/python3.8/site-packages/matplotlib/cbook/__init__.py:1402: FutureWarning: Support for mu
  ndim = x[:, None].ndim
/opt/conda/lib/python3.8/site-packages/matplotlib/axes/_base.py:276: FutureWarning: Support for multi-d
  x = x[:, np.newaxis]
/opt/conda/lib/python3.8/site-packages/matplotlib/axes/_base.py:278: FutureWarning: Support for multi-d
  y = y[:, np.newaxis]
```


KDE Contours of Casual vs Registered Riders
Daily Count Registered Riders
8000
6000
4000
2000
0
−2000
0
1000
2000
3000
4000
Daily Count Casual Riders

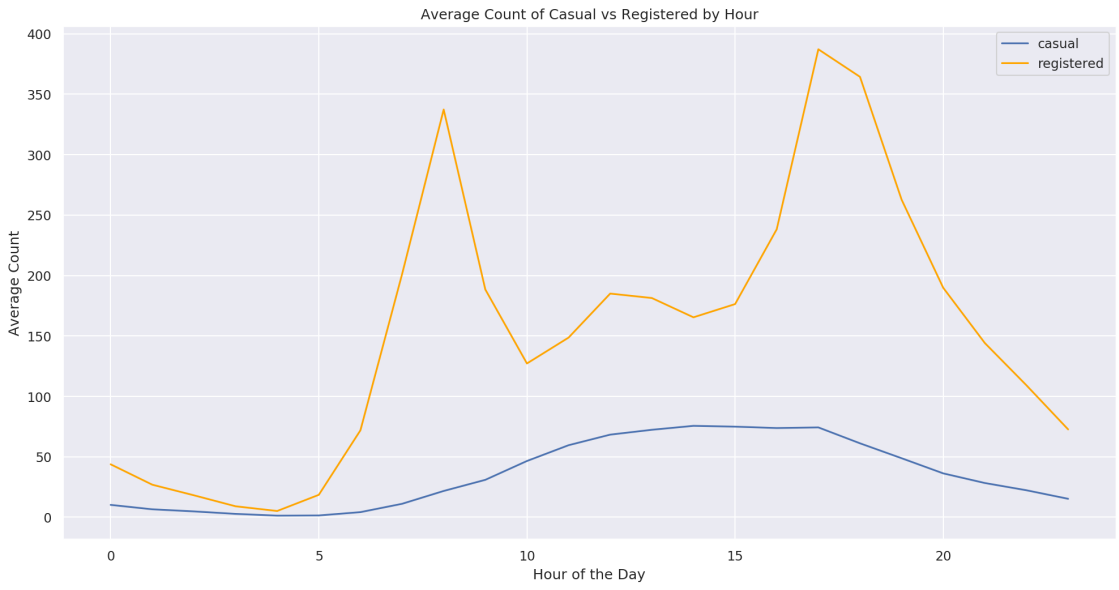
0.2 5: Understanding Daily Patterns

0.2.1 Question 5

Question 5a Let's examine the behavior of riders by plotting the average number of riders for each hour of the day over the **entire dataset**, stratified by rider type.

Your plot should look like the plot below. While we don't expect your plot's colors to match ours exactly, your plot should have different colored lines for different kinds of riders.

```
In [72]: avg_hour = bike.groupby("hr").agg({"casual": np.mean, "registered": np.mean})

         sns.lineplot(x='hr', y='casual',data= avg_hour, color='b')
         sns.lineplot(x='hr', y='registered',data= avg_hour, color='orange')

         plt.title('Average Count of Casual vs Registered by Hour')
         plt.xlabel("Hour of the Day")
         plt.ylabel("Average Count")
         plt.legend(['casual','registered'])
         plt.show()
```



Question 5b What can you observe from the plot? Hypothesize about the meaning of the peaks in the registered riders' distribution.

One major peice of information that we can observe from the graph is that there is always more registered users than casual bike users for every hour. The registered users seem to have peak registration at around 8 am and 4 pm (around rush hour.

In our case with the bike ridership data, we want 7 curves, one for each day of the week. The x-axis will be the temperature and the y-axis will be a smoothed version of the proportion of casual riders.

You should use statsmodels.nonparametric.smoothers_lowess.lowess just like the example above. Unlike the example above, plot ONLY the lowess curve. Do not plot the actual data, which would result in overplotting. For this problem, the simplest way is to use a loop.

You do not need to match the colors on our sample plot as long as the colors in your plot make it easy to distinguish which day they represent.

Hints: * Start by just plotting only one day of the week to make sure you can do that first.

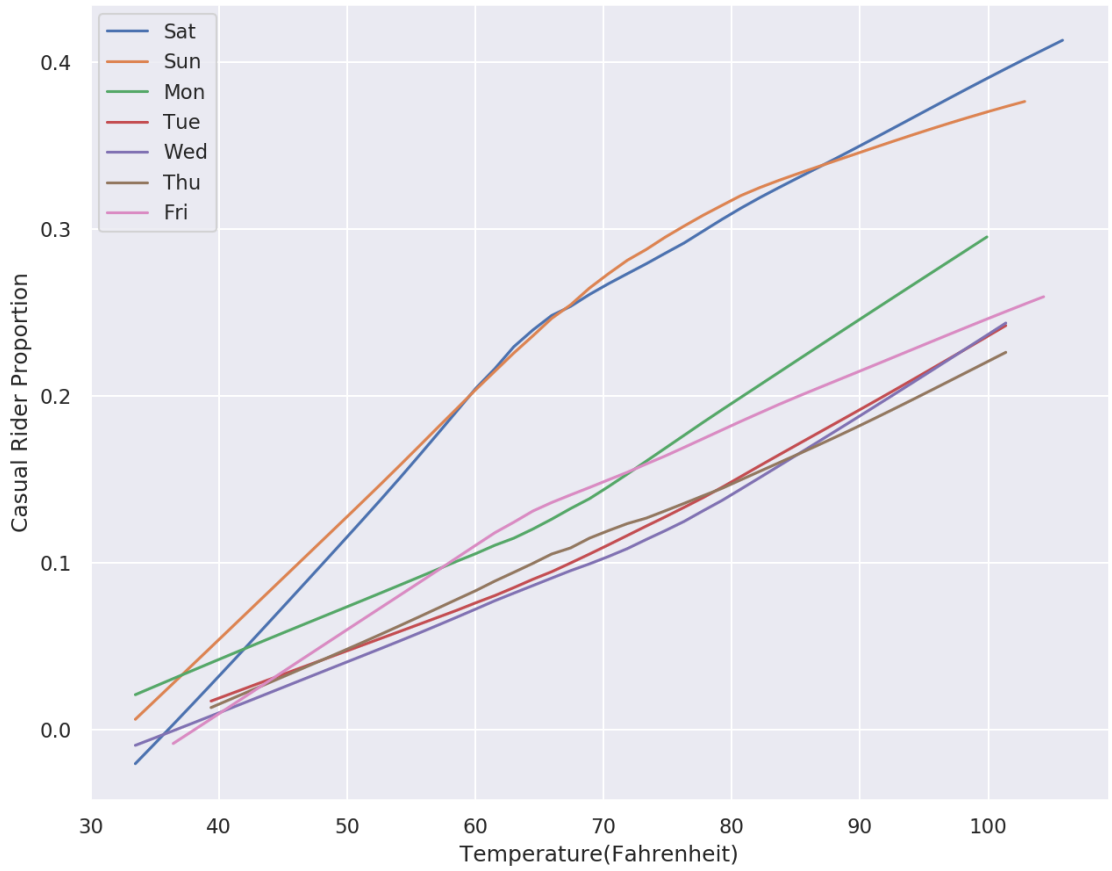
- The `lowess` function expects y coordinate first, then x coordinate.
- Look at the top of this homework notebook for a description of the temperature field to know how to convert to Fahrenheit. By default, the temperature field ranges from 0.0 to 1.0. In case you need it, Fahrenheit = Celsius $* \frac{9}{5} + 32$.

Note: If you prefer plotting temperatures in Celsius, that's fine as well!

In [82]:
```
from statsmodels.nonparametric.smoothers_lowess import lowess

plt.figure(figsize=(10,8))

weekdays = ['Sat','Sun', 'Mon', 'Tue', 'Wed', 'Thu', 'Fri']

for x in weekdays:
    weekday = bike[bike['weekday'] == x]
    weekday = lowess(weekday['prop_casual'].values, 9/5 * weekday['temp'].values * 41 + 32, re
    sns.lineplot(9/5*bike[bike['weekday'] == x]['temp']*41 + 32, weekday, label=x)

plt.xlabel('Temperature(Fahrenheit)')
plt.ylabel('Casual Rider Proportion')
plt.show()
```

```
/opt/conda/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following var
  warnings.warn(
/opt/conda/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following var
  warnings.warn(
/opt/conda/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following var
  warnings.warn(
/opt/conda/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following var
  warnings.warn(
/opt/conda/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following var
  warnings.warn(
/opt/conda/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following var
  warnings.warn(
```

```
/opt/conda/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following vari
  warnings.warn(
```



Question 6c What do you see from the curve plot? How is `prop_casual` changing as a function of temperature? Do you notice anything else interesting?

As the tempterature is increasing, the proportion of casual riders to overall number of riders rises. Also Saturday and Sunday have the highest proportion of casual riders comapared to other days in the week.

0.2.2 Question 7

Question 7A Imagine you are working for a Bike Sharing Company that collaborates with city planners, transportation agencies, and policy makers in order to implement bike sharing in a city. These stakeholders would like to reduce congestion and lower transportation costs. They also want to ensure the bike sharing program is implemented equitably. In this sense, equity is a social value that is informing the deployment and assessment of your bike sharing technology.

Equity in transportation includes: improving the ability of people of different socio-economic classes, genders, races, and neighborhoods to access and afford the transportation services, and assessing how inclusive transportation systems are over time.

Do you think the `bike` data as it is can help you assess equity? If so, please explain. If not, how would you change the dataset? You may discuss how you would change the granularity, what other kinds of variables you'd introduce to it, or anything else that might help you answer this question.

The data provided in bike is not enough to assess equity. Our data is more specific to count data, observing the weather and number of rentals made as a result. Equity is more a factor of socioeconomic class, gender, race, location -- information regarding the rental or renter which we do not have in the dataset. If we were to collect this data regarding our renters as well as costs of rentals, we can better address and assess equity.

Question 7B Bike sharing is growing in popularity and new cities and regions are making efforts to implement bike sharing systems that complement their other transportation offerings. The goals of these efforts are to have bike sharing serve as an alternate form of transportation in order to alleviate congestion, provide geographic connectivity, reduce carbon emissions, and promote inclusion among communities.

Bike sharing systems have spread to many cities across the country. The company you work for asks you to determine the feasibility of expanding bike sharing to additional cities of the U.S.

Based on your plots in this assignment, what would you recommend and why? Please list at least two reasons why, and mention which plot(s) you drew you analysis from.

Note: There isn't a set right or wrong answer for this question, feel free to come up with your own conclusions based on evidence from your plots!

Two major key correlated points that we saw in the graphs above are rents in relation to temperature and in relation to workdays. If a company were to open their business in new cities, they should prefer working cities with warmer temperatures which would increase the number of rentals per hour/per day. This would promote rentals due to the weather and allow people who need to travel to work an alternative option.