Use the `head` command on your three files again. This time, describe at least one potential problem with the data you see. Consider issues with missing values and bad data.

```
In [455]: bus.head()
```

Out[455]:

	business id column	name	address	city	state	postal_code	latitude	longitude	phone_number
0	1000	HEUNG YUEN RESTAURANT	3279 22nd St	San Francisco	CA	94110	37.755282	-122.420493	-9999
1	100010	ILLY CAFFE SF_PIER 39	PIER 39 K-106-B	San Francisco	CA	94133	-9999.000000	-9999.000000	14154827284
2	100017	AMICI'S EAST COAST PIZZERIA	475 06th St	San Francisco	CA	94103	-9999.000000	-9999.000000	14155279839
3	100026	LOCAL CATERING	1566 CARROLL AVE	San Francisco	CA	94124	-9999.000000	-9999.000000	14155860315
4	100030	OUI OUI! MACARON	2200 JERROLD AVE STE C	San Francisco	CA	94124	-9999.000000	-9999.000000	14159702675

```
In [456]: ins.head()
```

Out[456]:

	iid	date	score	type
0	100010_20190329	03/29/2019 12:00:00 AM	-1	New Construction
1	100010_20190403	04/03/2019 12:00:00 AM	100	Routine - Unscheduled
2	100017_20190417	04/17/2019 12:00:00 AM	-1	New Ownership
3	100017_20190816	08/16/2019 12:00:00 AM	91	Routine - Unscheduled
4	100017_20190826	08/26/2019 12:00:00 AM	-1	Reinspection/Followup

```
In [457]: vio.head()
```

Out[457]:

	description	risk_category	vid
0	Consumer advisory not provided for raw or unde…	Moderate Risk	103128
1	Contaminated or adulterated food	High Risk	103108
2	Discharge from employee nose mouth or eye	Moderate Risk	103117
3	Employee eating or smoking	Moderate Risk	103118
4	Food in poor condition	Moderate Risk	103123

There seem to be invalid or incorrect values in the dataset. For example, in bus, HeungYuen Restaurant's phone number is recorded as -9999, and the businesses from row [1:4] have negative values for latitude and longitude. There may also be human error, because HeungYuen Restaurant's ID number is 1000, which is of different format than the other Restaurant ID numbers. Furthermore, in inspections, certain restaurants have -1 as their score, which is also incorrect data. These may be default values for null values, and they must be standardized and cleaned.

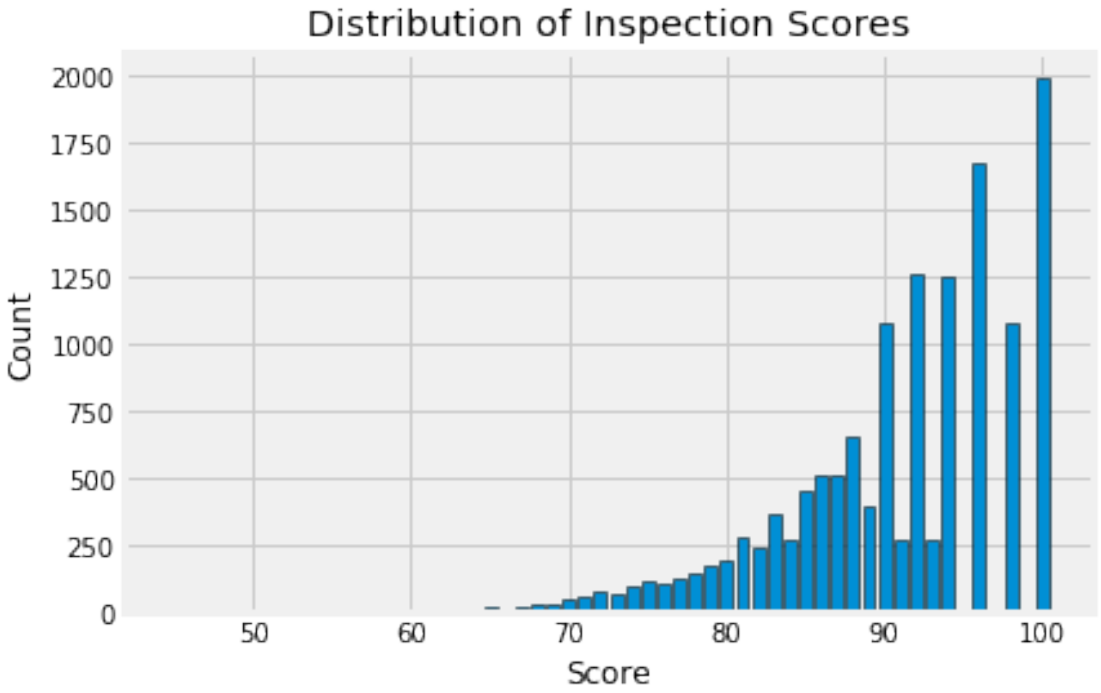
In the cell below, write the name of the restaurant with the lowest inspection scores ever. You can also head to yelp.com and look up the reviews page for this restaurant. Feel free to add anything interesting you want to share.

Lollipot; pretty good ratings on Yelp.

0.1 Question 6a

Let's look at the distribution of inspection scores. As we saw before when we called head on this data frame, inspection scores appear to be integer values. The discreteness of this variable means that we can use a barplot to visualize the distribution of the inspection score. Make a bar plot of the counts of the number of inspections receiving each score.

It should look like the image below. It does not need to look exactly the same (e.g., no grid), but make sure that all labels and axes are correct.



You might find this matplotlib.pyplot tutorial useful. Key syntax that you'll need:

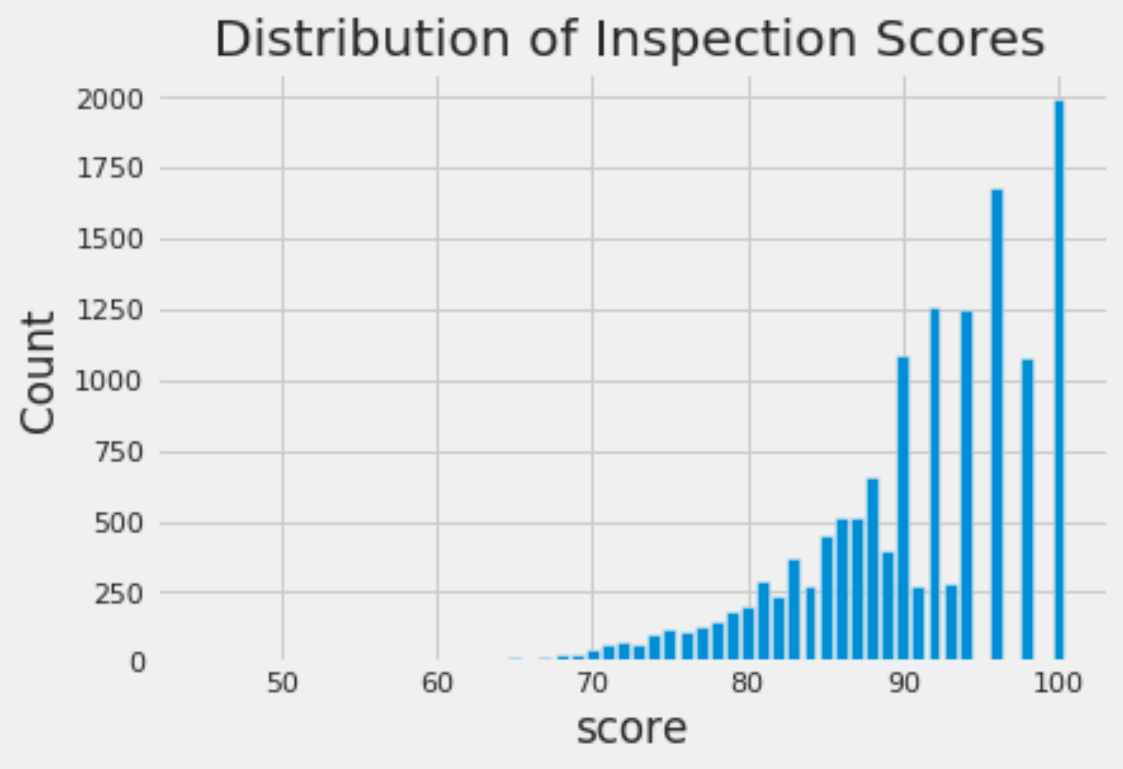
```
plt.bar
plt.xlabel
plt.ylabel
plt.title
```

Note: If you want to use another plotting library for your plots (e.g. plotly, sns) you are welcome to use that library instead so long as it works on DataHub. If you use seaborn sns.countplot(), you may need to manually set what to display on xticks.

```
In [533]: ins["score"].value_counts().sort_index().tail()
```

```
Out[533]: 93      277
          94     1250
          96     1681
          98     1080
          100    1993
          Name: score, dtype: int64
```

```
In [534]: counts = ins["score"].value_counts()
          plt.bar(x=counts.index, height=counts)
          plt.title("Distribution of Inspection Scores")
          plt.xlabel("score")
          plt.ylabel("Count");
```

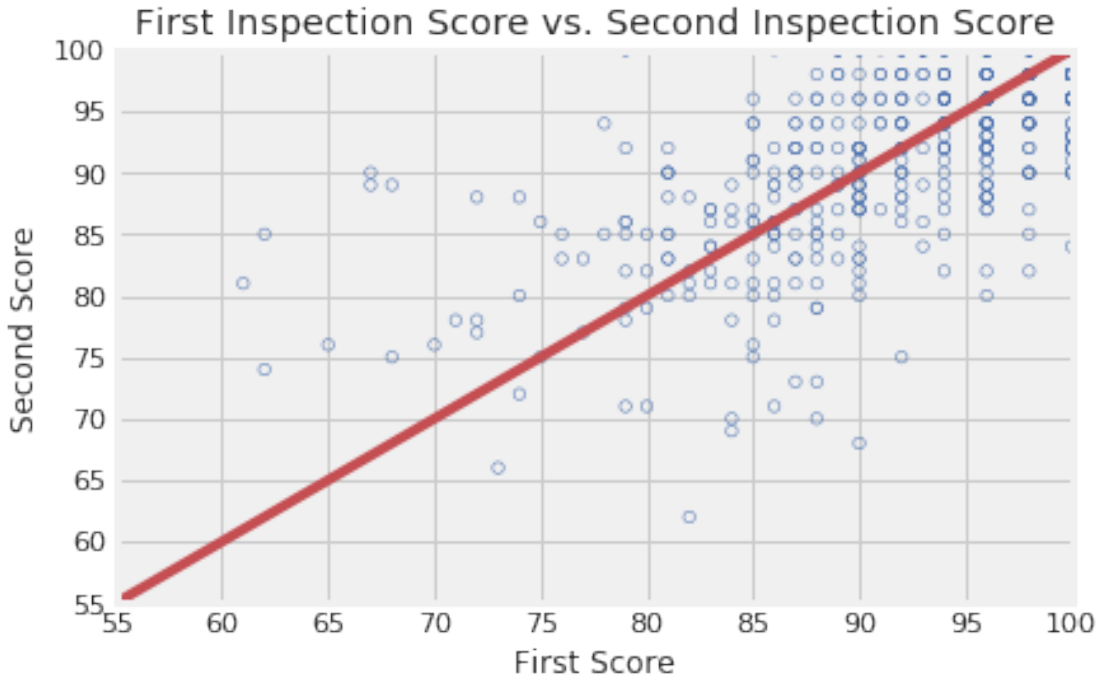


0.1.1 Question 6b

Describe the qualities of the distribution of the inspections scores based on your bar plot. Consider the mode(s), symmetry, tails, gaps, and anomalous values. Are there any unusual features of this distribution? What do your observations imply about the scores?

The mode of the bar graph is 100. There are gaps in the scores in the 90's, namely 99, 97, 95 -- odd sc

Now, create your scatter plot in the cell below. It does not need to look exactly the same (e.g., no grid) as the sample below, but make sure that all labels, axes and data itself are correct.



Key pieces of syntax you'll need:

`plt.scatter` plots a set of points. Use `facecolors='none'` and `edgecolors=b` to make circle markers with blue borders.

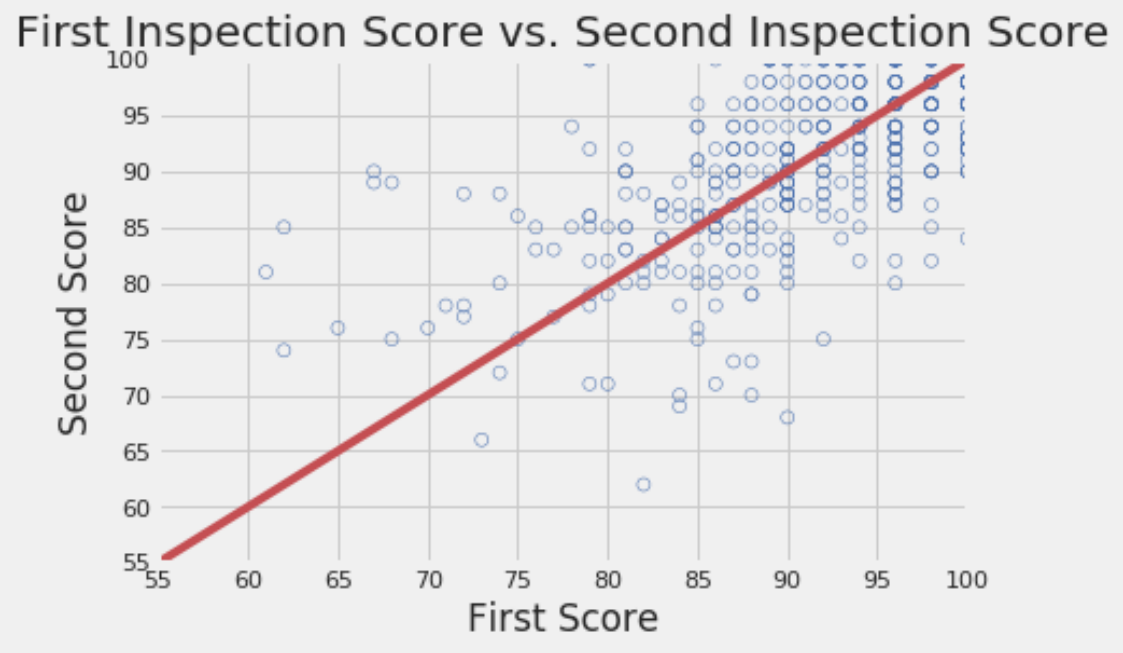
`plt.plot` for the reference line.

`plt.xlabel`, `plt.ylabel`, `plt.axis`, and `plt.title`.

Hint: You may find it convenient to use the `zip()` function to unzip scores in the list.

```
In [549]: coord = list(zip(*list(scores_pairs_by_business["score_pair"])))
          first_score = coord[0]
          second_score = coord[1]
```

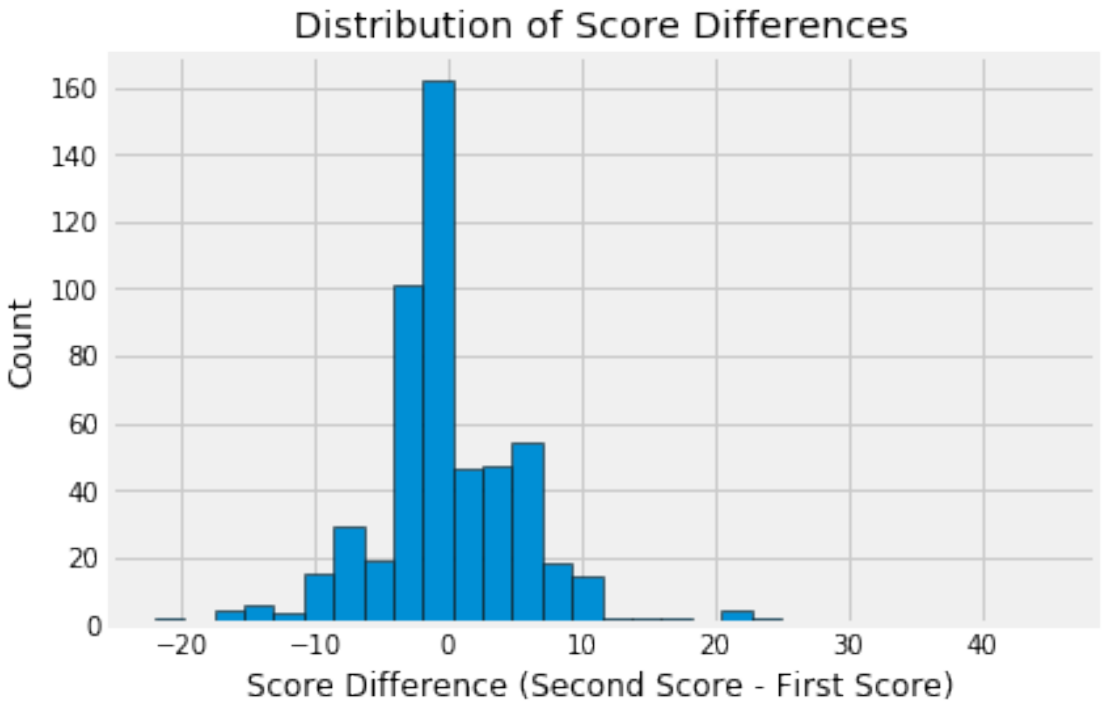
```
In [550]: plt.scatter(first_score, second_score, facecolors='none', edgecolors="b");
          plt.plot(list(range(55, 101, 1)), list(range(55, 101, 1)), 'r')
          plt.xlabel("First Score")
          plt.ylabel("Second Score")
          plt.title("First Inspection Score vs. Second Inspection Score")
          plt.axis([55, 100, 55, 100]);
```


First Inspection Score vs. Second Inspection Score
Second Score
First Score
100
95
90
85
80
75
70
65
60
55
55
60
65
70
75
80
85
90
95
100

0.1.2 Question 7d

Another way to compare the scores from the two inspections is to examine the difference in scores. Subtract the first score from the second in `scores_pairs_by_business`. Make a histogram of these differences in the scores. We might expect these differences to be positive, indicating an improvement from the first to the second inspection.

The histogram should look like this:



Hint: Use `second_score` and `first_score` created in the scatter plot code above.

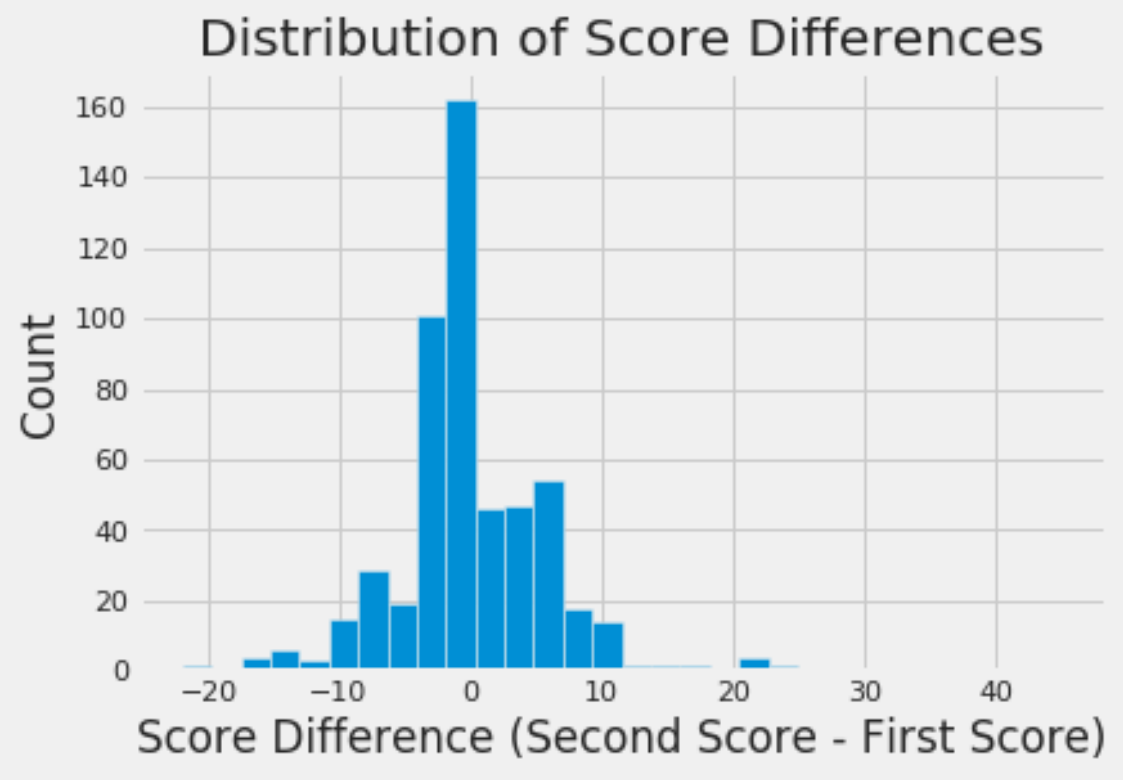
Hint: Convert the scores into numpy arrays to make them easier to deal with.

Hint: Use `plt.hist()` Try changing the number of bins when you call `plt.hist()`.

```
In [551]: diff = np.array(second_score) - np.array(first_score)
          np.median(diff), np.mean(diff), np.std(diff), max(diff)-min(diff)
```

```
Out[551]: (0.0, 0.30654205607476637, 6.225021428151197, 67)
```

In [552]:
```
plt.hist(diff, bins=30);
plt.xlabel("Score Difference (Second Score - First Score)")
plt.ylabel("Count")
plt.title("Distribution of Score Differences");
```



0.1.3 Question 7e

If restaurants' scores tend to improve from the first to the second inspection, what do you expect to see in the scatter plot that you made in question 7c? What do you oberve from the plot? Are your observations consistent with your expectations?

Hint: What does the slope represent?

If restaurants' scores tend to improve from the first to the second inspection, I expect the points to be above the reference line of slope 1. If the points were along the line, that would imply that there was no change in the scores from the first to second inspections. So if there was an improvement, the points would be above the line. My observations show that the scatter plot showed that the points were somewhat along the reference line, suggesting that there is no significant change from the first to the second scores.

0.1.4 Question 7f

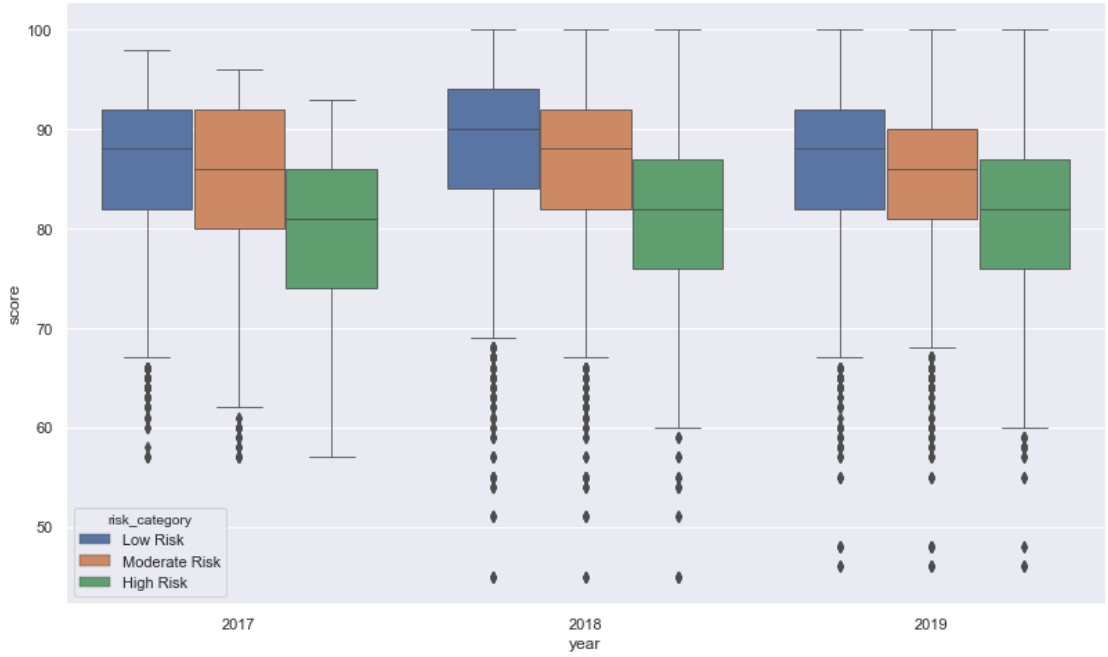
If a restaurant's score improves from the first to the second inspection, how would this be reflected in the histogram of the difference in the scores that you made in question 7d? What do you oberve from the plot? Are your observations consistent with your expectations? Explain your observations in the language of Statistics: for instance, the center, the spread, the deviation etc.

If a restaurant's score improves from the first to the second inspection, the histogram would be skewed to the left, with the center of the histogram (median / mean) being a positive value. My observations are not consistent with my expectations because the histogram is centered around 0. The mean of the differences is 0.31, and the median is 0, showing, in fact, that there is a slight right skew, and it does not the support the hypothesis that the inspection scores would improve. The standard deviation is 6.23, and the range of the data is 67.

0.1.5 Question 7g

To wrap up our analysis of the restaurant ratings over time, one final metric we will be looking at is the distribution of restaurant scores over time. Create a side-by-side boxplot that shows the distribution of these scores for each different risk category from 2017 to 2019. Use a figure size of at least 12 by 8.

The boxplot should look similar to the sample below. Make sure the boxes are in the correct order!



Hint: Use `sns.boxplot()`. Try taking a look at the first several parameters. The documentation is linked here!

Hint: Use `plt.figure()` to adjust the figure size of your plot.

```
In [553]: clean = merged7b[merged7b["year"]!=2016]
          clean.head()
```

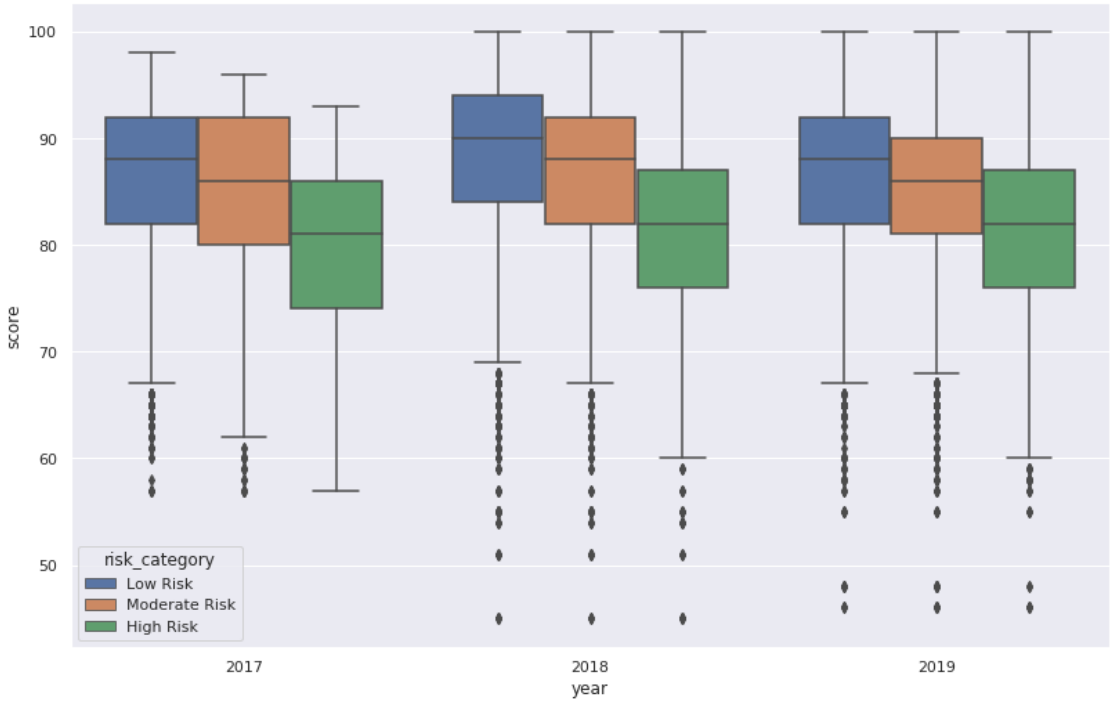
```
Out[553]:              iid                    date  score                    type  \
          0  100010_20190403  04/03/2019 12:00:00 AM    100  Routine - Unscheduled
          1  100017_20190816  08/16/2019 12:00:00 AM     91  Routine - Unscheduled
          2  100017_20190816  08/16/2019 12:00:00 AM     91  Routine - Unscheduled
          3  100041_20190520  05/20/2019 12:00:00 AM     83  Routine - Unscheduled
```

```
4  100041_20190520  05/20/2019 12:00:00 AM     83  Routine - Unscheduled

      bid  timestamp  year  Missing Score       vid               description  \
0  100010 2019-04-03  2019          False       NaN                       NaN
1  100017 2019-08-16  2019          False  103105.0  Improper cooling methods
2  100017 2019-08-16  2019          False  103139.0     Improper food storage
3  100041 2019-05-20  2019          False  103105.0  Improper cooling methods
4  100041 2019-05-20  2019          False  103139.0     Improper food storage

  risk_category
0           NaN
1     High Risk
2      Low Risk
3     High Risk
4      Low Risk
```

In [554]:
```
# Do not modify this line
sns.set();
plt.figure(figsize=(12, 8));
sns.boxplot(x= "year", y= "score" , hue='risk_category', data = clean, hue_order=["Low Risk",
```



---

1 8: Open Ended Question

1.1 Question 8a

1.1.1 Compute Something Interesting

Play with the data and try to compute something interesting about the data. Please try to use at least one of groupby, pivot, or merge (or all of the above).

Please show your work in the cell below and describe in words what you found in the same cell. This question will be graded leniently but good solutions may be used to create future homework problems.

1.1.2 Grading

Since the question is more open ended, we will have a more relaxed rubric, classifying your answers into the following three categories:

- **Great** (4 points): Uses a combination of pandas operations (such as groupby, pivot, merge) to answer a relevant question about the data. The text description provides a reasonable interpretation of the result.
- **Passing** (1-3 points): Computation is flawed or very simple. The text description is incomplete but makes some sense.
- **Unsatisfactory** (0 points): No computation is performed, or a computation with completely wrong results.

Please have both your code and your explanation in the same one cell below. Any work in any other cell will not be graded.

```
In [555]: ins2vio.head(5)
```

Out[555]:

	iid	vid
0	97975_20190725	103124
1	85986_20161011	103114
2	95754_20190327	103124
3	77005_20170429	103120
4	4794_20181030	103138

In [556]:
```
#YOUR CODE HERE
mult_ins2018 = ins[ins["year"]==2018].groupby("bid", as_index=False).filter(lambda sf: len(sf
merged = mult_ins2018.merge(ins2vio, how="left", on="iid")
merged = merged.merge(vio, how="left", on="vid")
merged = merged.sort_values("date")

high = merged.groupby(["bid", "risk_category"], as_index=False).filter(lambda sf: (sf["risk_ca
high_bid = high["bid"].unique()
lowmod = merged[~merged["bid"].isin(high_bid)]["bid"].unique()

def change(series):
    return series.iloc[0] - series.iloc[-1]

high_df = merged[merged["bid"].isin(high_bid)]
high_change = high_df[["bid", 'score']].groupby("bid", as_index=False).agg(change)
lowmod_df = merged[merged["bid"].isin(lowmod)]
lowmod_change = lowmod_df[["bid", 'score']].groupby("bid", as_index=False).agg(change)


# #YOUR EXPLANATION HERE (in a comment)
# I examined the the distributions of score improvements between restaurants that received an
# violation vs. restaurants that did not receive any "High Risk" violations in 2018. The merg
# is found by merging ins, ins2vio, and vio dataframes to include restaurants that received 2
# inspections in 2018, sorted by date.
# Then, "high" was found by grouping and filtering the "merged" dataframe to include any busi
# that had gotten at least 1 "High Risk" violation. Restaurants that had not received any "Hi
# violations are in "lowmod_df." The "change" function was used to find the change in scores
# the first to the last inspection of the year, under the assumption that there should be an
# in scores over the year.
```

```
  File "<ipython-input-556-887ed3ff95d8>", line 21
    I examined the the distributions of score improvements between restaurants that received any "Hi
    ^
SyntaxError: invalid syntax
```

1.1.3 Grading

Since the question is more open ended, we will have a more relaxed rubric, classifying your answers into the following three categories:

- **Great** (4 points): The chart is well designed, and the data computation is correct. The text written articulates a reasonable metric and correctly describes the relevant insight and answer to the question you are interested in.
- **Passing** (1-3 points): A chart is produced but with some flaws such as bad encoding. The text written is incomplete but makes some sense.
- **Unsatisfactory** (0 points): No chart is created, or a chart with completely wrong results.

We will lean towards being generous with the grading. We might also either discuss in discussion or post on Piazza some examplar analysis you have done (with your permission)!

You should have the following in your answers: * a few visualizations; Please limit your visualizations to 5 plots. * a few sentences (not too long please!)

Please note that you will only receive support in OH and Piazza for Matplotlib and seaborn questions. However, you may use some other Python libraries to help you create you visualizations. If you do so, make sure it is compatible with the PDF export (e.g., Plotly does not create PDFs properly, which we need for Gradescope).

In [581]:
```
# YOUR DATA PROCESSING AND PLOTTING HERE
high_change["category"] = 'high'
lowmod_change["category"] = 'lowmod'
score_diff = pd.concat([high_change, lowmod_change])

sns.displot(score_diff, x="score", kind="kde", hue="category")
plt.xlabel("Score Difference (Last Score - First Score)")
plt.ylabel("Count");
plt.title("Distribution of Score Differences");

# YOUR EXPLANATION HERE (in a comment)
# Used density curves because there was a huge peak at 0 for "lowmod" that made the "high" di
# Both distributions are roughly symmetrical and unimodal. The mean and median of the high ri
# businesses were around -1, showing on average a decrease in inspection
# scores, while the mean and median of the low/moderate risk businesses were around 0, showin
# significant change.
# However, the businesses that have not had any high risk violations show a much narrower spr
# with 0 being the mode of the distribution, showing a very high count of businesses that
# showed no change in score between inspections. This may be because of a high number of busi
# that scored 100. In contrast, the distribution of businesses with low to moderate risk viol
# does not have a distinct mode, and shows that most businesses' inspection scores did change
```

```
/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]
```