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.

One problem with the data in the Businesses dataframe is that for 4 rows out of the first 5, there are missing values of 'longitude' and 'latitude', and these missing values are represented by -9999. These also appear in 'phone_number' and they make it hard to study these particular restaurants because we are at risk of making our model inaccurate or biased, since they are not valid features of the dataset. A less glaring observation is one about 'business id column', where most Business IDs seem to be a six-digit number but the Business ID corresponding to Hueng Yuen Restaurant is a four-digit number. This may be an indication of bad/incomplete data because it is possible that the last two digits of the Business ID were never recorded.

A potential problem with the Inspections dataframe is that all of the times in the 'date' column seem to be 12:00:00AM which seems unlikely that the inspections were conducted at this time. Furthermore, a lot of values in 'score' are -1, which, again, seems unlikely to be an actual value for the score so it may represent either bad data or missing values.

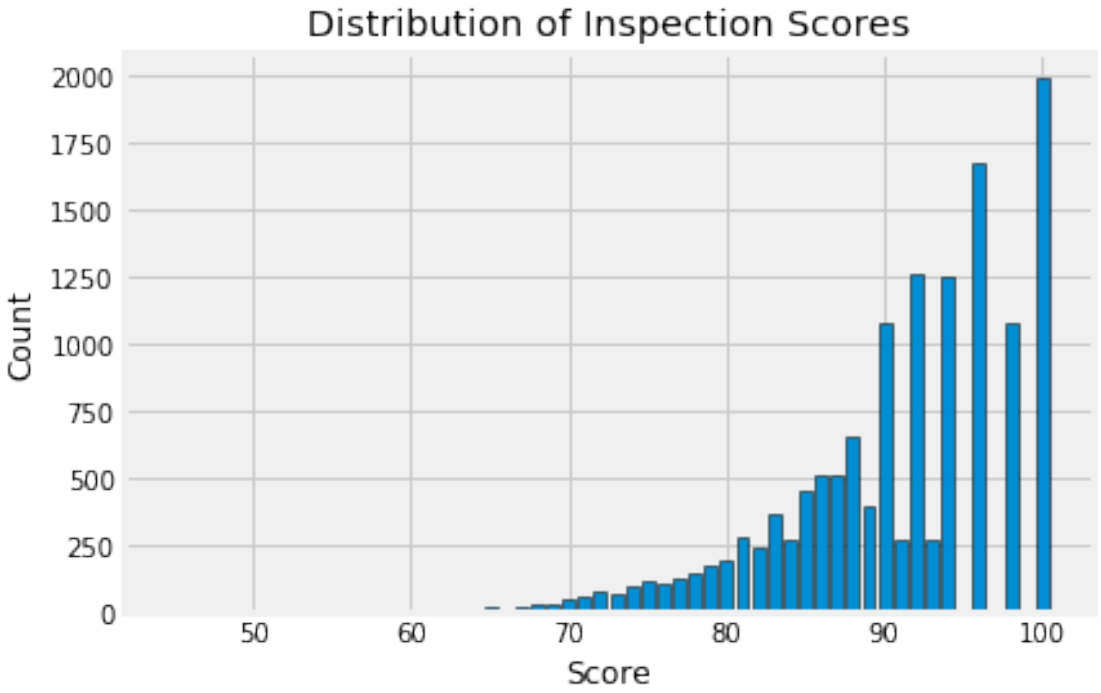
In the cell below, write the name of the restaurant with the lowest inspection scores ever. You can also head to yelp.com and look up the reviews page for this restaurant. Feel free to add anything interesting you want to share.

'Lollipot' was the restaurant with the lowest inspection scores ever. Something interesting I discovered while reading Yelp reviews for Lollipot was how much people disliked the restrooms there, but almost everyone thought the staff and service was friendly!

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 [161]:
```
plt.bar(ins['score'].value_counts().keys(), ins['score'].value_counts(), edgecolor='black')
plt.xlabel('Score', fontsize=13)
plt.ylabel('Count', fontsize=13)
plt.title('Distribution of Inspection Scores', fontsize=14)
```

Out[161]: Text(0.5, 1.0, 'Distribution of Inspection Scores')

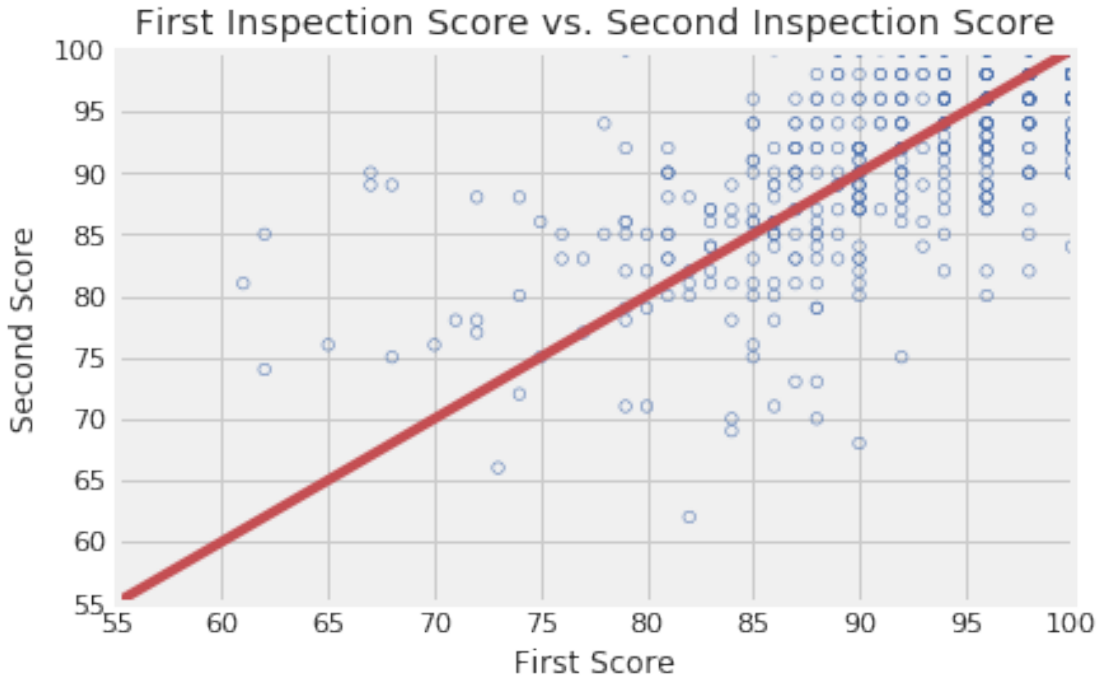
0.1.1 Question 6b

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 is 100, the graph is asymmetrical and left-skewed since there is a tail to the left because the mode occurs on the right side of the graph. There are significant gaps in the data because there are some scores that no restaurants received e.g. in the 99, 97...etc. These gaps become smaller for scores between 70-90, implying that restaurants have received almost all scores in this range. However, these gaps become larger as you go below 70. Anomalous values are those that would occur below a score of 70.

The unusual features of this distribution are that the mode is the maximum value, and there are (visually) very few scores below 65 which implies that restaurants with low scores do not survive in the competitive market and quickly go out of business, until only the best ones are left.

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.

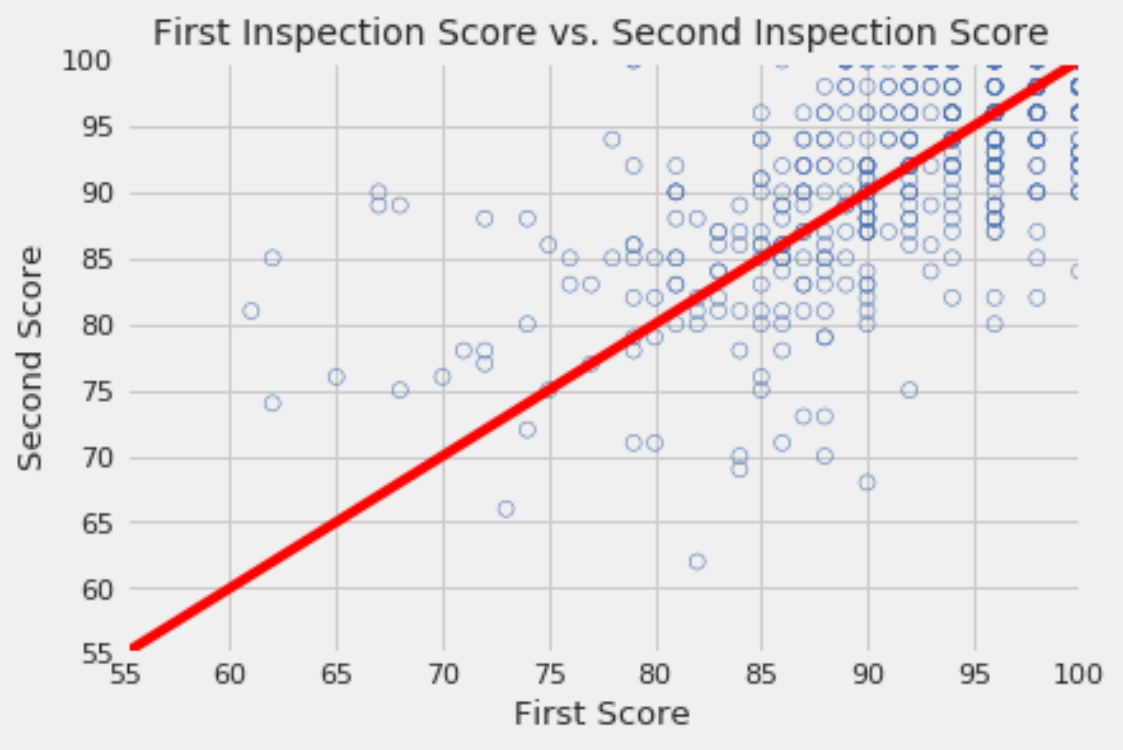
`plt.plot` for the reference line.

`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 [173]: first_score = [x[0] for x in scores_pairs_by_business['score_pair']]
          second_score = [y[1] for y in scores_pairs_by_business['score_pair']]
          plt.scatter(first_score, second_score, facecolors='none', edgecolors='b')
          plt.plot([55, 100], [55, 100], color='red')
          plt.axis(xmin=55, xmax=100, ymin=55, ymax=100)
          plt.title('First Inspection Score vs. Second Inspection Score', fontsize=14)
          plt.xlabel('First Score', fontsize=13)
          plt.ylabel('Second Score', fontsize=13)
```

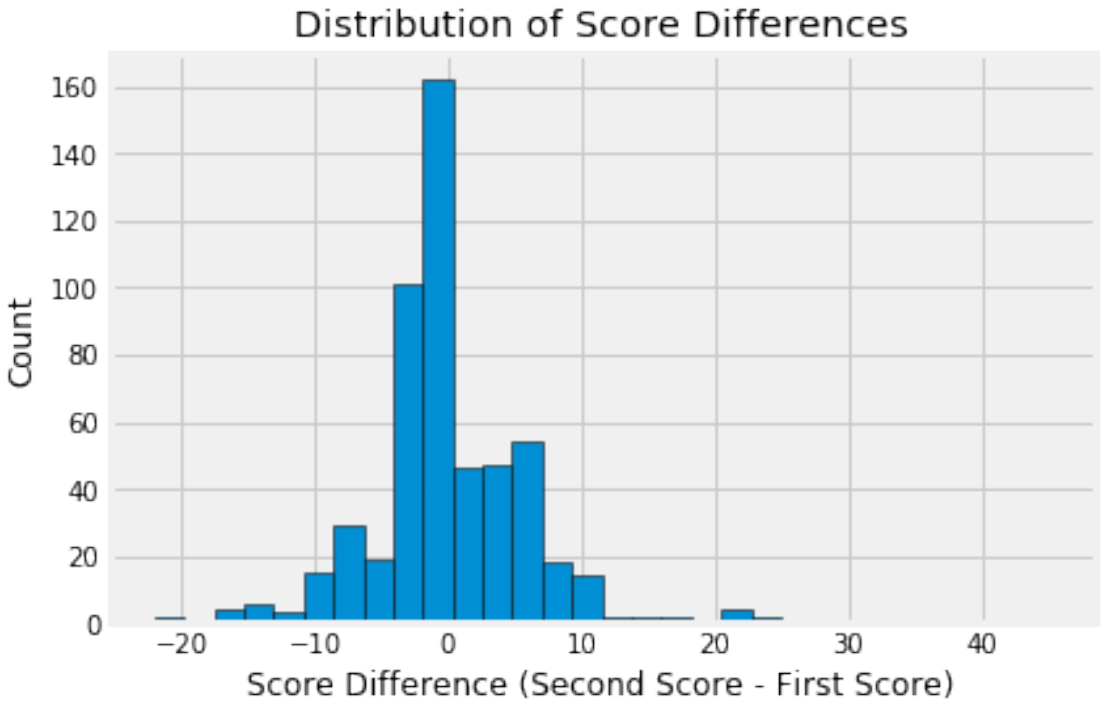
```
Out[173]: Text(0, 0.5, 'Second Score')
```


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.

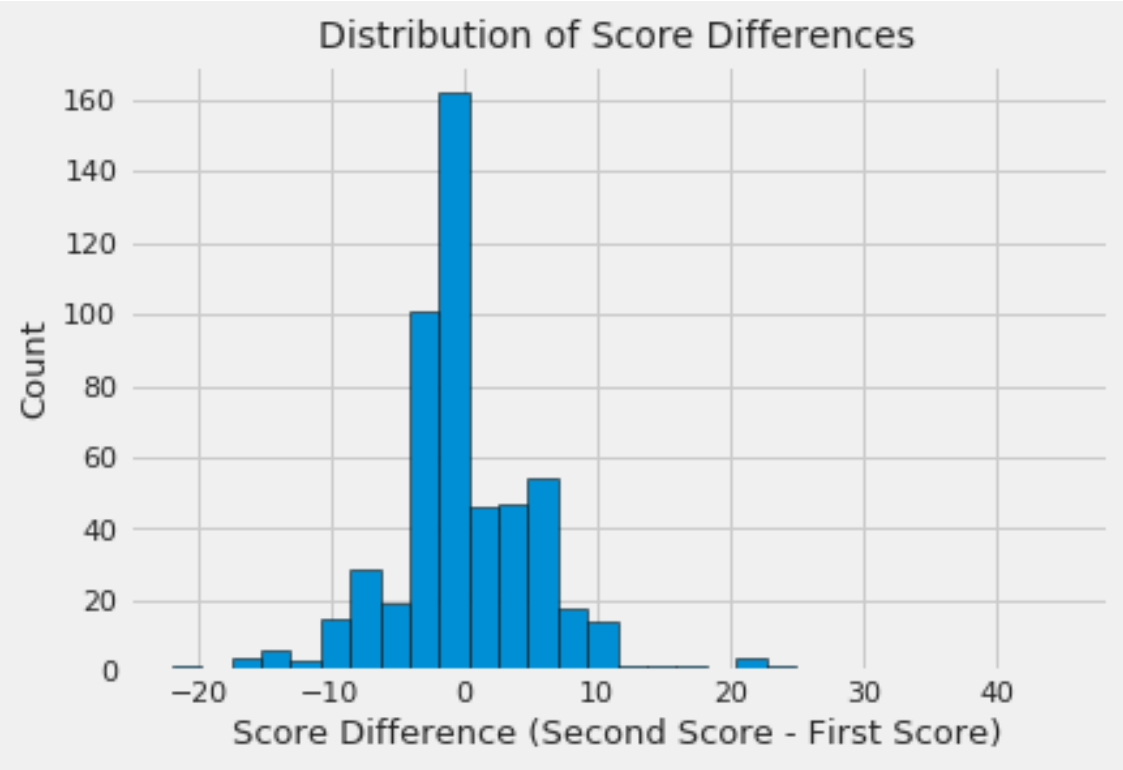
Hint: Convert the scores into numpy arrays to make them easier to deal with.

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

```
In [174]: first_score_array = np.array(first_score)
          second_score_array = np.array(second_score)
          score_difference = second_score_array-first_score_array
          plt.hist(score_difference, bins=30, edgecolor='black')
          plt.xlabel('Score Difference (Second Score - First Score)', fontsize=13)
```

```
plt.ylabel('Count', fontsize=13)
plt.title('Distribution of Score Differences', fontsize=14)
```

```
Out[174]: Text(0.5, 1.0, '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, this implies that more points will lie above the reference line than below. This is because the slope represents the ratio of second inspection scores to first inspection scores, and since that slope is 1, this means that it represents the restaurants whose score does not change from the first inspection to the second inspection. Points below the line have higher first inspection scores than second inspection scores.

However, my observations are not consistent with my expectations because according to the scatter plot, more points lie below the line than above. Although this difference is not stark, it is an indication that it is not necessarily the case that second inspection scores are better than first inspection 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.

This would be represented by a higher amount of values above zero, or simply an increase in non-negative values i.e. values to the right of zero. This could shift the center of the distribution to the right, but a guaranteed change would occur if enough restauarants' scores improve from the first to the second inspection.

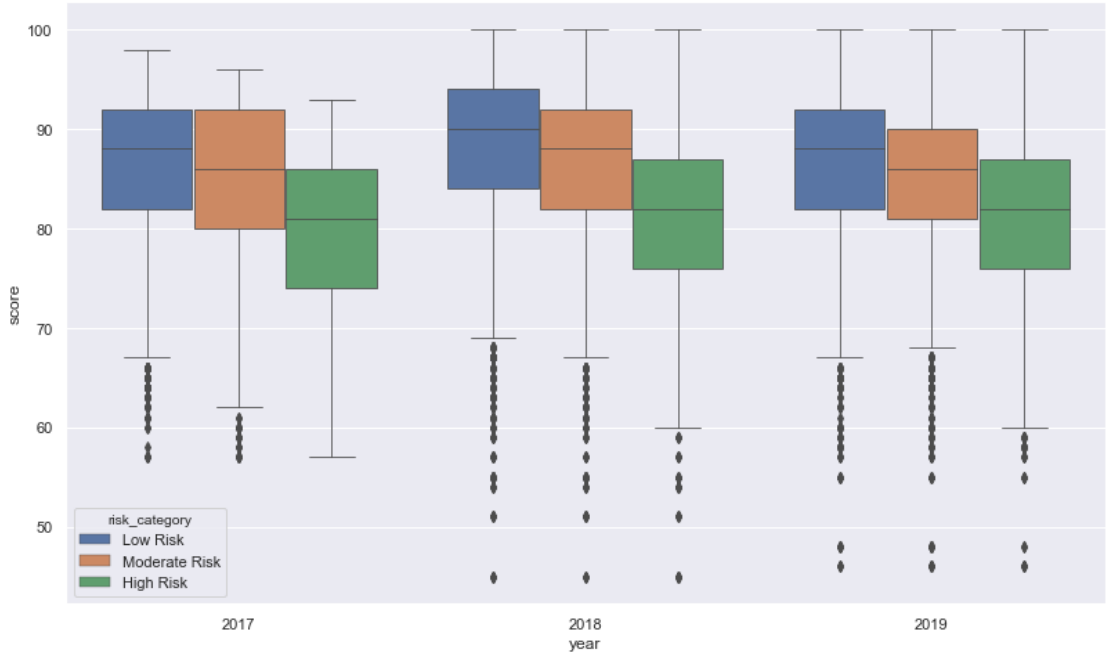
However, it is difficult to say anything about the spread and the standard deviation because we don't know the mean of this distribution. If it were a normal curve, we could look at it and say that the mean would occur at the axis of symmetry of the distribution, but since it's not, we cannot conclude that. If we don't know the mean, we can't say anything about the standard deviation because it is basically the deviation from the mean. For these reasons, we can't say anything about the spread either.

My observations are not consistent with my expectations because for most restaurants, the score difference from inspection 1 to inspection 2 is negative, which means that most restaurants got a lower score for inspection 2. Our expectation was that scores would improve, and this was not the case.

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 [175]:
```
# Do not modify this line

#change string to int
new_int_year = boxplot_data['year'].astype(int)
boxplot_data = boxplot_data.drop(columns={'year'})
boxplot_data['year']=new_int_year
#merge tables
boxplot_data = ins.merge(ins2vio, on='iid').merge(vio, on='vid')
boxplot_data = boxplot_data[boxplot_data['year']>2016]
```

```
  #plot
  sns.set()
  plt.figure(figsize = (12, 8))
  sns.boxplot(x='year', y='score', hue='risk_category', data=boxplot_data, hue_order={'Low Risk
```

```
---------------------------------------------------------------------------

NameError                                 Traceback (most recent call last)

<ipython-input-175-9a9c016e5a23> in <module>
  2
  3 #change string to int
----> 4 new_int_year = boxplot_data['year'].astype(int)
  5 boxplot_data = boxplot_data.drop(columns={'year'})
  6 boxplot_data['year']=new_int_year


NameError: name 'boxplot_data' is not defined
```

---

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 [176]:
```
#YOUR CODE HERE
risk_descriptions = pd.merge(vio, ins2vio, on='vid').merge(ins_named, on='iid').drop(columns=
risk_by_year = risk_descriptions.pivot_table(index='description', columns='year', values='bid
total = risk_by_year[2017]+risk_by_year[2018]+risk_by_year[2019]
risk_by_year['Total'] = total
risk_by_year_sorted = risk_by_year.sort_values(by=['Total'], ascending=False).reset_index().d
risk_by_year_sorted.head(5)

#YOUR EXPLANATION HERE (in a comment)
# The purpose of this is to analyze how violations have changed over the years, e.g. some vio
```

```
# unapproved utensils grew in number from 2017 to 2019, while others such as risk vermin infe
# number through these years. The purpose is to analyze which ones grew, and to associate glo
# intermingling with changing hygiene consciousness with their rise/decline.

# Each row of this table contains one violation and the number of times that violation occure
# 2019. I have dropped the column for 2016 because 2016 has very few inspection records compa
# three years, so including it would cause my study to be inaccurate. For simplicity, I will
# most common violations.
```

Out[176]:

year	description	2017	2018	2019
0	Unclean or degraded floors walls or ceilings	1018	1016	888
1	Unapproved or unmaintained equipment or utensils	700	889	967
2	Inadequately cleaned or sanitized food contact…	688	726	1091
3	Inadequate and inaccessible handwashing facili…	686	815	860
4	Moderate risk food holding temperature	665	726	861

year	Total
0	2922
1	2556
2	2505
3	2361
4	2252

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 [177]:
```
# YOUR DATA PROCESSING AND PLOTTING HERE
risk_by_year_sorted.iloc[0:20].plot(kind='barh', x='description', y={2017,2018,2019}, ylabel=
# YOUR EXPLANATION HERE (in a comment)
# The graph below shows the description of the 21 most common violations and the number of ti
# 2017 (showed in blue), the number of times they occurred in 2018 (showed in orange) and the
# occurred in 2019 (showed in green). The violations that have increased over time are those
# green bar is larger than the orange and blue bars, and the violations that have decreased o
# which the blue bar is larger than the orange and green bars.

# According to our data, the violation that has most obviously increased is "Inadequately cle
# contact surfaces". The violation that has most obviously decreased is "Moderate risk vermin
# "Unclean or degraded walls or ceilings". It is interesting to note that most of the violati
# had to do with the absence of a utilities e.g.  hot running water, or improper cooling meth
# temperatures. This may be the result of a PG&E price hike in 2019 which made it harder for
# afford expensive utility bills.
# On the other hand, most of the violations that decreased had to do with a physically cleane
# e.g. less vermin infestations etc. which was probably the result of customers scrutinizing
# standards more and by the popularization of Yelp, Tripadvisor etc, which pressurizes restau
# I'm interested to see how this data changes due to COVID next year!
```

Out[177]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0b5d4467c0>
```


Violation by Year
year
2017
2018
2019
Violation
High risk vermin infestation
Permit license or inspection report not posted
Insufficient hot water or running water
Inadequate food safety knowledge or lack of certified food safety manager
Improper cooling methods
Improper storage of equipment utensils or linens
Low risk vermin infestation
Food safety certificate or food handler card not available
Unclean or unsanitary food contact surfaces
Unclean nonfood contact surfaces
High risk food holding temperature
Improper food storage
Moderate risk vermin infestation
Foods not protected from contamination
Wiping cloths not clean or properly stored or inadequate sanitizer
Moderate risk food holding temperature
Inadequate and inaccessible handwashing facilities
Inadequately cleaned or sanitized food contact surfaces
Unapproved or unmaintained equipment or utensils
Unclean or degraded floors walls or ceilings
0
200
400
600
800
1000