**Pandas Class Practice :)**

1. **Create two dataframes using the dictionaries below then merge them using inner, outer, right, left join**

Car\_Price = {'Company': ['Toyota', 'Honda', 'BMV', 'Audi'], 'Price': [23845, 17995, 135925 , 71400]}

car\_Horsepower = {'Company': ['Toyota', 'Honda', 'BMV', 'Audi'], 'horsepower':

[141, 80, 182 , 160]}

**NOTE:**

When to use the Pandas concat vs. merge and join While merge, join, and concat all work to combine multiple DataFrames, they are used for very different things. In this section, we’ll learn when you will want to use one operation over another. The key distinction is whether you want to combine your DataFrames horizontally or vertically.

The concat method allows you to combine DataFrames vertically.

Imagine you had two DataFrames with the same columns. Perhaps the first DataFrame includes 10 rows of stock trading data for one stock while the second DataFrame includes 10 rows of stock trading data for a different stock. A vertical combination would use a DataFrame’s concat method to combine the two DataFrames into a single DataFrame with twenty rows.

Notice that in a vertical combination with concat, the number of rows has increased but the number of columns has stayed the same

By contrast, the merge and join methods help to combine DataFrames horizontally.

Imagine you have two DataFrames. The first contains stock trading information various companies. The second contains information about the headquarters and numbers of employees for a particular company. If the two DataFrames have one field in common—such as a stock symbol or company name—you can combine the two DataFrames so that each row contains both the stock trading data and the company background information.

Notice that in this horizontal combination, we aren’t adding any additional rows. Rather, we’re adding columns to existing rows. The horizontal combination from a merge operation is similar to a JOIN operator in SQL.

1. **Create two data frames using the dictionaries below, Concatenate those two data frames and create a key for each data frame**

Hint syntax for concat : new\_dataframe = pd.concat([dataframe1, dataframe2], keys=['key1', 'key2'])

GermanCars = {'Company': ['Ford', 'Mercedes', 'BMV', 'Audi'], 'Price': [23845, 171995, 135925 , 71400]}

japaneseCars = {'Company': ['Toyota', 'Honda', 'Nissan', 'Mitsubishi '], 'Price': [29995, 23600, 61500 , 58900]}

1. **Read the CSV file shared on Slack**
2. **Replace all column values which contain ‘?’ and 'n.a' and 'null' with NaN (Hint : use the read\_csv argument called na\_values)**
3. **Create a new column called 'CC' by multiplying horsepower by 15**
4. **return the first 18 cars with a body-style called sedan**
5. **Create a copy of your dataframe(Hint use copy())**
6. **Using loc, access all the values for columns company, body-style, CC and price**
7. **Using loc, return row 18, 25 and 60**
8. **Using iloc, access all the values for columns company, body-style, CC and price**
9. **Using iloc, return row 18, 25 and 60**
10. **Use corr() function to find the correlation among the columns in the dataframe using ‘Pearson’ method.**
11. **Use corr() function to find the correlation among the columns in the dataframe using ‘kendall’ method.**
12. **What is the correlation between average-mileage and price?**

Pearson -0.770217

Kendall -0.716072

1. **Print All alpha-romero Cars details**
2. **Whats the maximum price in the dataset?**
3. **Which is the most expensive car company name?**
4. **Count the total number of cars each company has. ( hint value\_counts() )**
5. **What is the highest priced car for each company? ( hint first group by the company and save it in a variable then check for the maximum price from the saved variable )**
6. **What is the average mileage for each car making company**