**Ecommerce Purchases Exercise**

In this Exercise you will be given some Fake Data about some purchases done through Amazon! Just go ahead and follow the directions and try your best to answer the questions and complete the tasks. Feel free to reference the solutions. Most of the tasks can be solved in different ways. For the most part, the questions get progressively harder.

Please excuse anything that doesn't make "Real-World" sense in the dataframe, all the data is fake and made-up.

Also note that all of these questions can be answered with one line of code.

\*\* Import pandas and read in the Ecommerce Purchases csv file and set it to a DataFrame called ecom. \*\*

**Check the head of the DataFrame.**

\*\* How many rows and columns are there? \*\*

\*\* What is the average Purchase Price? \*\*

\*\* What were the highest and lowest purchase prices? \*\*

\*\* How many people have English 'en' as their Language of choice on the website? \*\*

\*\* How many people have the job title of "Lawyer" ? \*\*

\*\* How many people made the purchase during the AM and how many people made the purchase during PM ? \*\*

\*\* What are the 5 most common Job Titles? \*\*

\*\* Someone made a purchase that came from Lot: "90 WT" , what was the Purchase Price for this transaction? \*\*

\*\* What is the email of the person with the following Credit Card Number: 4926535242672853 \*\*

\* How many people have American Express as their Credit Card Provider \*and made a purchase above $95 ?\*\*

\*\* Hard: How many people have a credit card that expires in 2025? \*\*

\*\* Hard: What are the top 5 most popular email providers/hosts (e.g. gmail.com, yahoo.com, etc...) \*\*

# SF Salaries Exercise

Welcome to a quick exercise for you to practice your pandas skills! We will be using the [SF Salaries Dataset](https://www.kaggle.com/kaggle/sf-salaries) from Kaggle! Just follow along and complete the tasks outlined in bold below. The tasks will get harder and harder as you go along.

\*\* Import pandas as pd.\*\*

\*\* Read Salaries.csv as a dataframe called sal.\*\*

\*\* Check the head of the DataFrame. \*\*

\*\* Use the .info() method to find out how many entries there are.\*\*

**What is the average BasePay ?**

\*\* What is the highest amount of OvertimePay in the dataset ? \*\*

\*\*  What is the job title of JOSEPH DRISCOLL ? Note: Use all caps, otherwise you may get an answer that doesn't match up (there is also a lowercase Joseph Driscoll). \*\*

\*\* How much does JOSEPH DRISCOLL make (including benefits)? \*\*

\*\* What is the name of highest paid person (including benefits)?\*\*

\*\* What is the name of lowest paid person (including benefits)? Do you notice something strange about how much he or she is paid?\*\*

\*\* What was the average (mean) BasePay of all employees per year? (2011-2014) ? \*\*

\*\* How many unique job titles are there? \*\*

\*\* What are the top 5 most common jobs? \*\*

\*\* How many Job Titles were represented by only one person in 2013? (e.g. Job Titles with only one occurence in 2013?) \*\*

\*\* How many people have the word Chief in their job title? (This is pretty tricky) \*\*

\*\* Bonus: Is there a correlation between length of the Job Title string and Salary? \*\*

# 911 Calls Capstone Project

For this capstone project we will be analyzing some 911 call data from [Kaggle](https://www.kaggle.com/mchirico/montcoalert" \t "_blank). The data contains the following fields:

* lat : String variable, Latitude
* lng: String variable, Longitude
* desc: String variable, Description of the Emergency Call
* zip: String variable, Zipcode
* title: String variable, Title
* timeStamp: String variable, YYYY-MM-DD HH:MM:SS
* twp: String variable, Township
* addr: String variable, Address
* e: String variable, Dummy variable (always 1)

Just go along with this notebook and try to complete the instructions or answer the questions in bold using your Python and Data Science skills!

\*\* Import numpy and pandas \*\*

\*\* Import visualization libraries and set %matplotlib inline. \*\*

\*\* Read in the csv file as a dataframe called df \*\*

\*\* Check the info() of the df \*\*

\*\* Check the head of df \*\*

\*\* What are the top 5 zipcodes for 911 calls? \*\*

\*\* What are the top 5 townships (twp) for 911 calls? \*\*

\*\* Take a look at the 'title' column, how many unique title codes are there? \*\*

## Creating new features

\*\* In the titles column there are "Reasons/Departments" specified before the title code. These are EMS, Fire, and Traffic. Use .apply() with a custom lambda expression to create a new column called "Reason" that contains this string value.\*\*

\*For example, if the title column value is EMS: BACK PAINS/INJURY , the Reason column value would be EMS. \*

\*\* What is the most common Reason for a 911 call based off of this new column? \*\*

\*\* Now use seaborn to create a countplot of 911 calls by Reason. \*\*

\*\* Now let us begin to focus on time information. What is the data type of the objects in the timeStamp column? \*\*

\*\* You should have seen that these timestamps are still strings. Use [pd.to\_datetime](http://pandas.pydata.org/pandas-docs/stable/generated/pandas.to_datetime.html" \t "_blank) to convert the column from strings to DateTime objects. \*\*

\*\* You can now grab specific attributes from a Datetime object by calling them. For example:\*\*

time = df['timeStamp'].iloc[0]

time.hour

**You can use Jupyter's tab method to explore the various attributes you can call. Now that the timestamp column are actually DateTime objects, use .apply() to create 3 new columns called Hour, Month, and Day of Week. You will create these columns based off of the timeStamp column, reference the solutions if you get stuck on this step.**

\*\* Notice how the Day of Week is an integer 0-6. Use the .map() with this dictionary to map the actual string names to the day of the week: \*\*

dmap = {0:'Mon',1:'Tue',2:'Wed',3:'Thu',4:'Fri',5:'Sat',6:'Sun'}

\*\* Now use seaborn to create a countplot of the Day of Week column with the hue based off of the Reason column. \*\*

**Now do the same for Month:**

**Did you notice something strange about the Plot?**

\*\* You should have noticed it was missing some Months, let's see if we can maybe fill in this information by plotting the information in another way, possibly a simple line plot that fills in the missing months, in order to do this, we'll need to do some work with pandas... \*\*

\*\* Now create a gropuby object called byMonth, where you group the DataFrame by the month column and use the count() method for aggregation. Use the head() method on this returned DataFrame. \*\*

\*\* Now create a simple plot off of the dataframe indicating the count of calls per month. \*\*

\*\* Now see if you can use seaborn's lmplot() to create a linear fit on the number of calls per month. Keep in mind you may need to reset the index to a column. \*\*

\*Create a new column called 'Date' that contains the date from the timeStamp column. You'll need to use apply along with the .date() method. \*

\*\* Now groupby this Date column with the count() aggregate and create a plot of counts of 911 calls.\*\*

\*\* Now recreate this plot but create 3 separate plots with each plot representing a Reason for the 911 call\*\*

\*\* Now let's move on to creating heatmaps with seaborn and our data. We'll first need to restructure the dataframe so that the columns become the Hours and the Index becomes the Day of the Week. There are lots of ways to do this, but I would recommend trying to combine groupby with an [unstack](http://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.unstack.html" \t "_blank) method. Reference the solutions if you get stuck on this!\*\*

\*\* Now create a HeatMap using this new DataFrame. \*\*

\*\* Now create a clustermap using this DataFrame. \*\*

\*\* Now repeat these same plots and operations, for a DataFrame that shows the Month as the column. \*\*