**Linear Regression assignment- Demand prediction:** *(Use the ‘****Demand\_txn’*** *dataset for the below tasks)*

* Read the dataset, print last 3 observations and find the data types of each column
* Convert the data type of ‘Mapped\_Item\_Code’ field to object
  + Hint: use **.astype(str)** for the data type conversion
* Convert the field ‘Transaction\_Date’ to datetime (Date) column
  + Hint: use **pd.to\_datetime** for conversion
* Extract only those records whose transaction date is greater than or equal to ‘**2017-01-01**’ and proceed with below operations
  + Hint: use **.loc** operator
* Check the changed data types of each column and also see how many unique sales types are available in ‘Mapped\_Sales\_Type’ column
* Extract only those records whose ‘Avg\_Discount\_Percent\_On\_Discounted\_Items’ is lesser than 1.0 and do a sense check if the resulting data frame contains any values in the same column that are greater than or equal to 1.0
* Create the below dictionary (dict\_agg) and pass it as an argument for the aggregate function in order to roll up the results grouped by following order: 'City', 'Mapped\_Sales\_Type', 'Mapped\_Item\_Code', 'Transaction\_Date'
  + dict\_aggr = {'Quantity\_Sold':'sum', 'Median\_Price':'median', 'Effective\_Price':'median'}
  + sample syntax: **df.groupby(<groupby list>).agg(dictionary)**
* Print first 5 observations where Effective\_Price is not null
* Print first 10 observations where ‘City’ is ‘Hyderabad’ **AND** ‘Mapped\_Sales\_Type’ is ‘Delivery’
  + Hint: **df.loc[(<condition 1>) & (<condition 2>), :].head(10)**
* Print first 3 observations where Percent\_Quantity\_With\_Discount is not equal to zero
  + Hint: use **!=** operator
* Create a new column ‘Range’ and populate it as **1** if Effective\_Price is greater than **500**, else **0.5**
  + Hint: df[‘New column’] = np.where(<condition>, if true, if false)
* Split the dataset into X and Y based on below list:

**x =** 'Mapped\_Item\_Code', 'Median\_Price', 'Average\_Tax\_Percent', 'Effective\_Price', 'Percent\_Quantity\_With\_Discount', 'Avg\_Discount\_Percent\_On\_Discounted\_Items','Car\_sales', 'F\_and\_B\_Inflation', 'F\_and\_B\_Inflation\_Adjusted\_Effective\_Price', 'F\_and\_B\_Inflation\_Adjusted\_Median\_Price'. And **y =** ‘Quantity\_Sold’

* Run the Linear Regression model on the above x and y features. Feel free to explore the modeling parameters and come up with a high R2
* While fitting x and y to get stats summary, data type of x should be converted to float due to categorical features, otherwise the summary would return error
  + Hint: sm.OLS(y, x.astype(float)).fit()