# Customer Data Pre-processing

# Background

You have been given a collection of data from a company wishing to process its customer records for business purposes (**acw\_user\_data.csv**). The existing systems in-place at the company only export to a CSV file, and this is not in an appropriate format for analysis. You have been given the task of preparing this data for further analyses by your colleagues within the company, including representation changes, filtering, and deriving some new attributes / metrics for them.

These data include attributes such as first name, second name, credit card number, marital status, and even contains data on the customer’s car.

The number of records provided is significant, and therefore it is expected that solutions are robust to varying types of data, and varying values, offering a programmatic solution.

# Tasks

## Data Processing (70%)

Using standard python (No pandas / seaborn) with default libraries (os, sys, time, json, csv,

…) you have been given the following tasks:

1. Read in the provided ACW Data using the CSV library.
2. As a CSV file is an entirely flat file structure, we need to convert our data back into its rich structure. Convert all flat structures into nested structures. These are notably:
   1. Vehicle - consists of make, model, year, and type
   2. Credit Card - consists of start date, end date, number, security code, and IBAN.
   3. Address - consists of the main address, city, and postcode.

For this task, it may be worthwhile inspecting the CSV headers to see which data columns may correspond to these above.

Note: Ensure that the values read in are appropriately cast to their respective types.

1. The client informs you that they have had difficulty with errors in the dependants column. Some entries are empty (i.e. “ “ or “”), which may hinder your conversion from Task 2. These should be changed into something meaningful when encountered.

Print a list where all such error corrections take place.

E.g. Problematic rows for dependants: [16, 58, 80, 98]

1. Write all records to a **processed.json** file in the JSON data format. This should be a list of dictionaries, where each index of the list is a dictionary representing a singular person.
2. You should create two additional file outputs, **retired.json** and **employed.json**, these should contain all retired customers (as indicated by the retired field in the CSV), and all employed customers respectively (as indicated by the employer field in the CSV) and be in the JSON data format.
3. The client states that there may be some issues with credit card entries. Any customers that have more than 10 years between their start and end date need writing to a separate file, called **remove\_ccard.json**, in the JSON data format. The client will manually deal with these later based on your output. They request that you write a function to help perform this, which accepts a single row from the CSV data,

and outputs whether the row should be flagged. This can then be used when determining whether to write the current person to the **remove\_ccard** file.

1. You have been tasked with calculating some additional metrics which will be used for ranking customers. You should create a new data attribute for our customers called “Salary-Commute”. Reading in from **processed.json**:
   1. Add, and calculate appropriately, this new attribute. It should represent the Salary that a customer earns, per mile of their commute.
      1. Note: If a person travels 1 or fewer commute Km, then their salary- commute would be just their salary.
   2. Sort these records by that new metric, in ascending order.
   3. Store the output file out as a JSON format, for a **commute.json** file.

## Data Visualization

Using Pandas and Seaborn

Your client wishes to understand the data they have on their customers a bit more by use of visualisations. With use of Pandas and Seaborn read in the original CSV file provided with the assignment.

1. Obtain the Data Series for Salary, and Age, and calculate the following:
   1. Mean Salary
   2. Median Age
2. Perform univariate plots of the following data attributes:
   1. Age, calculating how many bins would be required for a bin\_width of 5.
   2. Dependents, fixing data errors with seaborn itself.
   3. Age (of default bins), conditioned on Marital Status
3. Perform multivariate plots with the following data attributes:
   1. Commuted distance against salary.
   2. Age against Salary
   3. Age against Salary conditioned by Dependants
4. Your client would like the ability to save the plots which you have produced. Provide a Notebook cell which can do this.

## Code Presentation

The code you produce to solve the above tasks should make good use of structure, logic, and commenting to be clear and robust. Variable names should be thoughtfully considered, and appropriate for purpose. Ensure that scope conflicts are avoided, and that variable names don’t leak into other areas of code. You should consider how the structures covered throughout the module may be used. Similarly, you should be mindful of error handling where appropriate.

Use of Markdown Cells is advised to keep clear distinction between Tasks. When writing your algorithms for solving the above, it may be useful to keep in mind reusability of code, reducing the amount of boilerplate code, and duplicated code