Script process

Cost summaries are first copied into a folder so that the script isn’t too intensive in scanning over the entire internal drive each time the operation is completed.

Next is a for loop which loops over every worksheet in every spreadsheet. A function ‘get data’ is used to iterate over every row in each worksheet which retrieves the data into pandas Data Frames. A function called ‘staff extract generator’ is used is called, which drops empty cells and applies the squeeze nan function; this reformats the Data Frames to restructure a summarising ‘total costs’ cell to follow a heading ‘staff cost’. Between the heading and total costs in where the data lies. As these ‘labels’ are not consistently applied due to either error or process changes, the labels are modified to be uniform. Whitespace is stripped from the index as a series, and using lambda functions the labels are changed. Lambda is able to approximate the cell labels, therefore we can use this approximation as an if function, if the cell contains the pattern we want, which translates to a ‘staff’ heading or ‘total staff costs’ then it is changed to s total costs and staff costs. We can then add the edited index back into the Data Frame index and loc can be applied to specify the data required.

The filename and worksheet is then extracted while the spreadsheets are being manipulated, which will be utilized as a reference point, allowing us to locate the associated client in our main systems.

The data we require is only in the first four columns of the data set, and the spreadsheets may contain notes which would corrupt this data set, therefore the first four rows and created reference points are extracted from each worksheet.

As the spreadsheets we are iterating over contain worksheets that do not include the data set we require, a try and except function is applied, and the script prints the exception in case there are any worksheets that do contain our desired data that are not extracted due to errors.

Next is the ‘concatenated df’ function which creates a single Data Frame from each worksheet Data Frame. We can then begin to give shape to the final dataset. The column names are titled, including ‘r&d cost’ which is included in the original set but not included in the final Data Frame; but it is included at this point in case, although unlikely, a spreadsheet included additional information, i.e. the jo role and project role.

A reindex is used so that we can refine the job titles used in the staff costs. A regex is applied which identifies entries in the job titles column with text only, as columns which would have numeric values would class as an entry without a job title or staff name. At earlier stages of our company’s process in treating these costs we have not included the staff member’s name, and in this process these entries are eliminated. Although they are entries that contain apportionments and costs they don’t include the job role, which is vital, and I would consider this data less reliable. A similar validation is used against the apportionment and total cost. A lambda is applied to both columns which ensures that they are int or float types only. Strings are not desired. As the entries might be entered incorrectly, MAX and MIN functions are applied to extract the cost and apportionment respectively, under the assumption that the greatest figure will be the total cost, the second figure will be the calculation between the total cost and apportionment (not required) and the apportionment is the lower figure. The apportionment must also be between 0.1 and 100 to be considered a percentage of total costs. If there are any nan cells after this process, then the cells are dropped.

With the data set correctly formatted, and the bulk of data in correct order, the cif (ID) of the correlating client can be associated with the database of client data. The cif is not always in the filename, therefore the best way to extract the cif is from the data source, from our internal servers using ‘client id’ function. A Data Frame is constructed with file destinations including costs summary or cost summary as the data was originally extracted. A merge is completed between the Data Frame of file destinations and the staff costs dataset based on filenames. To ensure that the original index is not lost during the merge operation, a reindex is used and the job titles is ultimately used as the index. Based on the foldernames extracted merged into the Data Frame, a regex is utilized to extract the correlating cif number. As many clients have three to five digit numbers in their company names, the regex excludes these recognised digits in the Data Frame. File names might also include the correlating periods including 2013-2017 or expressed as 14-17, therefore these periods have also been excluded. If more than one cif has been extracted from these file destinations, then the function ‘tidy cif’ is used to split the digits extracted into separate rows. The cif column is then cleaned to remove characters created by the lists and remove empty cells, and their associated rows.

The year is then added from the worksheet of each spreadsheet. This might include the periods between certain dates or with a year, therefore a regex is used to identify the last two digits before the “] characters used in entering the worksheet names, and the ultimate two digits from this created dataset is extracted, ensuring that if two dates are extracted then the final two digits are the digits that represent the year-ending period. At each stage the created series’ have a function applied to remove the list formatting to return only the two digits we require.

The remaining step in cleaning the data is to remove duplicates, which is applied by subsetting the names, cif and worksheet, assuming there are no two individuals with the same name in the same period acting on behalf of our clients. The columns we need for the final dataset are selected, the cost, percentage, cif and year.

With the cost DataFrame obtained, it can be connected to an SQL database which includes the client data. The connection is made to the sql databases, which includes a database of client ID, but there is also a database with claim statuses. This needs to be extrapolated so that we can specify the data of clients which have been submitted and paid for, the process completed entirely. There will be a lot of data for which clients have been cancelled, or haven’t paid, or the claims haven’t been submitted yet. This data set also needs the yearly periods isolated, which utilizes two possibilities, that the year periods are at the end of the cell, or are defined by two hyphens. As there are only two errors in the columns, these rows are dropped, as they do not contribute significantly to the data set but do create problems. The final columns are selected from the merged data set. This is made into a CSV file.

The subcontractor databases is completed with key differences to the staff costs. The labels used to define the location of the data required for this data set is set to xsubcontractor and xtotal sub to create unique identifiers for the data. The columns are not selected until after the worksheets are concatenated and an additional column is added. The additional column in needed as the subcontractors have an additional column to detail whether they’re connected to the client or not – if they’re not connected only 65% of their costs are claimable. Consequently, this dataset is more vulnerable to shifted columns, where a column will have a part of another column’s data. Due to this vulnerability in the data set, the max and min functions used to unscramble the data is more important. The consumable and software data sets are created in the same manner as the subcontraction costs to ensure consistency in the code, but with corresponding labels to extract relevant data.