# Data Converter

Using python, write me a tool to universally convert one data format into another. The input file can be either excel files or CSV files. This tool needs to be compatible with both format.

There should be a mapping file mapping original column names to new column names. The mapping file is in xlsx format, tab name is “column mapping”. This tab should contain 4 columns. The first column lists out all the column names in the original data file; the second column list out or the column names in the new data file; the third column list out all the data type in the new data file; the fourth column indicates whether the column in the new format is optional

There is no restriction on the input file column names but in the new file, all the column headers needs to be in uppercase, datatypes are also in uppercase, it can be INTEGER, BIGDEC for float, TEXT

There should be a config.yaml file containing all the configurations which include the file path to the input data file. And file path to the new data file.

The python code needs to be modularised for flexibility purpose

Put all the required packages into requirements.txt file

Set up the code structure and some sample input and output files as well as the mapping file for me. This is insurance company data, so in the sample you want to include columns like inception date, expiry date, building construction type, building fire protection factors, and so on

——————————————————————————————————

In my input file there could be multiple columns of fire protection factors such as smoke alarm, fire extinguisher, fire blanket. These columns will be in the form of flags with values 1 or 0. In the new format these multiple columns will be coerced into a list, so in the mapping file or three column names will be mapped into the same column name called fire protection and the data type should be ‘LIST’, the list should contain the column name as values whenever the flag is one, implement this in the code

——————————————————————————————————

When there’s a list in the output, it needs to be exploded with each value in a separate columns, and all other columns should be NA, except for the policy number column, which should fill down

——————————————————————————————————

Sorry not all NA, values in the other columns should remain there for the first row, then for the second and below rows when list expanded, they should be NA

——————————————————————————————————

The previous “whole list as text string” format is also useful, create a new parameter in the config.yaml to specify which format to use - “list\_in\_single\_string” or “list\_in\_multiple\_rows”.

When “list\_in\_single\_string” is specified, instead of “[item1,item2,item3]”, the value should present as this format: “item1;item2;item3”, if empty list, then it will be empty string “”

——————————————————————————————————

When format is “list\_in\_multiple\_rows”, if empty list, then it will be empty string “” instead of “[]”

——————————————————————————————————

i've updated my mapping\_column.xlsx to rename the data types:

- TEXT to CODE (validation rules: all capital letter, no longer than 12 chars including space)

- INTEGER to INTEGE, validation rules: raise error if longer than 12 digits

- BIGDEC to DECIMA: float with max 4 decimal places and max 12 digit long (total number of digits on both sides of decimal point), round to 4 dp if longer, and raise error if total length (exclusive of the decimal point) is longer than 12

new types:

- DATE: text in the format of 'dd/mm/yyyy', convert to this format if data provided was integer, use Excel origin date

new column:

- List: indicate whether a field is part of a list, this flag replaces the LIST type, each value in a list should follow the data type provided in the previous column

——————————————————————————————————

# Impact analysis

Now i want to generate an impact analysis report based on this output, I’ve created a new config file for this - “config\_impact\_analysis.yaml”, in there:

I’ve specified a mapping file with a “sheet\_input” parameter indicating the tab containing the following data:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Benchmark File** | **Benchmark Column** | **Target File** | **Target Column** |
| POLICY\_NUMBER | impact\_analysis/input/file1.xlsx | PREMIUM\_AMOUNT | impact\_analysis/input/file2.xlsx | PREMIUM\_AMOUNT |

The input files required for this exercise are in “Benchmark File” and “Target File” columns, and the columns needed for assessing the impact are specified in the “Benchmark Column” and “Target Column” columns, the values specified in the same row are to be compared against each other, i.e. You want compare values in column “PREMIUM\_AMOUNT” from the target file against column “PREMIUM\_AMOUNT” in the benchmark file. Your task is to import the files specified, only keep the first row for each value in the column specified in the “ID” column, in this case, “POLICY\_NUMBER”, then merge all the data based on their policy number, keep all unique columns that’s not used for comparison from all files (drop duplicate columns with the same names); for the comparison columns, add prefix using the file name (without extension)

Now from the tab specified in “sheet\_segment” in the config file, get the list of columns to aggregate the comparison columns by, then compute the percentage difference, which is derived by “Target Column / Benchmark Column - 1”, column name should be “diff\_” + column name.

Then in the tab specified in “sheet\_band”, map the difference computed above to the band name, then plot a bar chart using highchart, output a html file in the output dir in the config file

Also output the merged date file to the output folder

——————————————————————————————————

1. Modularise the approach for the impact analysis and data conversion
2. Use polars instead of pandas
3. I changed how i use the segment field. In the bar chart, y axis should be the proportion of policy, x axis should be bands, the segment field is used as filters, for each field in the segment list, create separate tabs for each unique value, then filter the data using the segment field’s values, plot one bar chart for each tab

——————————————————————————————————

arrange my files for the converter function, adopt a modularised approach

——————————————————————————————————

refactor the visualizer module to use the Jinja2 package, save the template in a separate file, and make this visualizer flexible by importing the components created by analyzer and chart\_generator

——————————————————————————————————

in chart generator, the bar chart should be vertical bars (maybe this is another chart type i can't remember), x axis labels should be the band names, y-axis should be the proportion, and put the x axis labels in the original order as specified in the mapping file, do not sort

——————————————————————————————————

now i have added a second column in file1 and file2 for comparison, and added a new row in the impact\_analysis\_mapping file indicating that this column also needs comparison. I've also added a column 'Item' to indicate what this comparison is named. you need to treat all columns specified in the mapping\_column tab in the mapping file as target and benchmark column, fix the data processing code for the merging, and need to calculate percentage difference for each comparison item, and there needs to have multiple column charts, each for the distribution for each comparison item, create tabs (arranged in a horizontal bar) for users to click through the charts, same apply to the summary tables, take the value from 'Item' as the tab names

——————————————————————————————————

in the band\_distribution tab, the band column needs to have the union of all bands for all comparison item, and it should have a pair of the 'count' and 'percentage' columns for each pair of comparison, put the comparison name as prefix to distinguish the columns

——————————————————————————————————

In data\_processor.py: the overall logic should be:

1. Get all benchmark and target columns from all files, and rename them first
2. Get all column names from all files and get the union set
3. Merge all files and keep unique columns in the union set, when there are common columns, keep the columns from the first file only
4. Calculate differences for each comparison item specified in the mapping file

Clean up the code using the logic above

# Dashboard app

Create a dash dashboard in Python which serves as a front end for 2 functions:

Page 1 - Allow users to import data and perform the data conversion.

In this page, there is a zone that allows users to drag in their import data. Give a preview of the data for the first few lines.

Secondly, the page should allow users to drag in a mapping file and give a preview of the output format.

Thirdly, there is a zone for users to preview the current config, which connects to config\_data\_conversion.yaml. Users can edit and update through front end.

Then it should have a button for users to start converting.

There should also be a log screen to show users what is happening and any error messages

Page 2 - Show the impact summary and chart

Similarly, there is a zone for users to preview the current config, which connects to config\_impact\_analysis.yaml. Users can edit and update through front end.

This app should utilise as much existing modules / code as possible through import

——————————————————————————————————

I got 18 errors with the following message:

Attempting to connect a callback Input item to component: "run-impact-analysis" but no components with that id exist in the layout. If you are assigning callbacks to components that are generated by other callbacks (and therefore not in the initial layout), you can suppress this exception by setting `suppress\_callback\_exceptions=True`. This ID was used in the callback(s) for Output(s): impact-analysis-status.children

Can you fix this? And you can use my browser to test any errors after fixing

——————————————————————————————————

In impact\_analysis backend, use the python package of highchart instead of js strings, and put the charts functions into a separate module so that it’ll be useful in dash callback, as well as in static report generation

——————————————————————————————————

# To Do

* Check the logic in the data processing for impact analysis
  1. Rename should happen first
  2. Then the merge logic should be simple