# File Structure

This universal mapping program was created in order to map any excel workbook to any database table.  
The program will walk you through the process, and provides robust error catching for known errors in current processes. The primary working directory that this program is designed to work in can be seen or set in the self.path variable in the class initializer. Variable definitions will be provided via inline comments.  
  
Each process is contained in a folder that is in the base directory (self.path), and you can add a new process by following the same folder structure  
example:  
Universal Mapping/  
├──Cap Leak/  
├──Cap Payments/  
├──Cumulative Roster/  
├──Roster/  
├──State Payments/  
├──<Example Process>/  
 ├──<Year\*\*>/  
 ├──<Workbook Type/ MCO\*\*>/  
 ├──Output/  
 ├──Mapping File.xlsx  
\*\*This structure can be adjusted on the [[Meta]](#_File_Path) tab of the mapping file, and is just an example. Files without '\*\*' need to be in the relative position shown and exist.  
  
All the current mappings follow this format. There are three tabs on each mapping. The first is [Mapping] which contains the mapping for the workbook. Second is the [Meta] tab, which contains metadata regarding the mapping, including things like sheet number (to collect data from) File path where the data can be located, database table to be inserted into, the method used to insert, header row, and archive paths. Finally, there is the [Date Input] tab, which contains the month and year that you would like to run the process for. This mapping assumes that the mapping that you would like to run follows these conventions, format, and dimensions.  
  
The mapping program also allows you to concatenate, split, hardcode values, and force datatype conversions through a simple syntax in the mapping file. This makes it so you do not have to hardcode processes in python.

The source code can be found [here](Universal%20Mapping.py).

# How to Run:

## Setting up Python / IDE

To run, you must have Python 3.7 or greater installed on your computer in order to run the program. To run in PyCharm, open the file in PyCharm, install the required dependencies, configure the interpreter, and click “Run”.

[PyCharm Link - Installation Guide](https://www.jetbrains.com/help/pycharm/installation-guide.html)

[Installing Dependencies Help](https://www.jetbrains.com/help/pycharm/installing-uninstalling-and-upgrading-packages.html)

[Configure interpreter in PyCharm](https://www.jetbrains.com/help/pycharm/configuring-python-interpreter.html)

Unfortunately this cannot be done through the command line due to policies to protect CHAS computers, and must be done through an IDE of some sort. I recommend PyCharm Community Edition since it is free, or Visual Studio Code (also free).

## Running Program

The program will walk you through the steps as you run it. It will ask you questions and require responses to move forward with the processes, as well as overrides when necessary. Make sure to review the summary of the data after mapping which will be printed after the mapping is complete to ensure that data integrity has not been lost. You can see what has been done before uploading to the database by exporting to excel in order to look at what has been mapped. The program will prompt you to do this before uploading to SQL.

# Column Operations:

## Concatenation (using the '+' operator):

To concatenate two columns, follow the format of <column name>+<column name>. Other examples include:

<column name>+<column name>+<column name>  
 <column name>+,+<column name>

You can choose to concatenate a string in between the columns if you would like, for example:

Member\_Last\_Name+,+Member\_First\_Name

Would take the columns Member\_Last\_Name and Member\_First\_Name and concatenate them together with a comma in the middle like Smith,John. Note: whitespaces will be concatenated as well.

## Splits (using the '|' operator): To split a column, follow the format of <column name>|<character to split on>|<index>. Example:

Member Name|,|0

Which would yield from the value 'Smith, John' --> 'Smith' or:

Member Name|,|1

Which would yield from the value 'Doe, Jane' --> 'Jane'

## Fixed Values (using the '\*' operator):

To create a fixed value in a column, all that is needed is to add the \* character to the end of the fixed value you would like to add on the end of the string you'd like to use as a fixed value. for example:

Molina\*

Which would yield a column that every value for every row contained the string 'Molina'. You can even use a calculated column in excel and add this to the end of the calculated\concatenated column to create a value that is fixed in the final table. example:

='Date Input'!$B$2&"/1/"&'Date Input'!$B$3&"\*"

Which would yield a fixed date based on the input on the [Date Input] tab of the workbook.

# Forced Datatyping (For now, just dates, datetimes, and stings):

On every mapping, there is a column that is called DataType which is the first column on every mapping. This column can be used to force the datatype of the finished column to a datetime (with time stamp) or just a date (without time stamp). For timestamped datetimes, type datetime in that column. For dates, type date into that column. To force a column to format as a string, type string into that column.

# Error Catching: There are multiple areas for error checking, although not all errors will be caught. Be sure to read the error carefully and try to use the solutions provided on errors where a suggested solution is provided. If there is an uncaught error, save the error as well as the raw file(s) for debugging later.

An example of a mapping error will provide you with a similar message to what is shown below:

Text

Description automatically generated

This message is telling you the MCO that is being mapped, the column which the file is being mapped to, and the mapped column. This allows for quick debugging as the column that was not found that was expected to be found is listed as the mapped column. Often when the column that comes up as an error is the first non-static column, this is because the header setting in the [[Meta]](#_File_Path) tab is set incorrectly.

# [Mapping] tab:

Each folder has its own mapping which you will need to visit to update the year. As long as this file is not auto saved enabled, you may leave the file open when you run the program and make changes to the file and save them as long as the program is not actively trying to access the map while you're saving.

Maps are created by first making a column called Upload Field which mirrors the columns in the table you would like to upload to, then creating columns the correspond to a file that would be mapped to this database.

Table

Description automatically generated

Here is an example of a map ^^^

Notice the special syntax used to split, force datatypes, and use static values. Mapping columns from the source file is done by adding the name of the column to be mapped with any special operations that need to be completed on that column. In the example the column bearing a header of PCP Term Date from the Amerigroup source file would be mapped to the PCPTermDate column in the SQL Table. Blank cells will show as NULL in the final SQL table.

# [Meta] tab:

The [[Meta]](#_File_Path) tab contains information about the file itself, and some instructions related to the file. Each file will have its own meta column which will have multiple rows. In this matrix you can control where files are archived, what sheet to use off the file, what row the header is in, and what type of method you would like to use when inserting into the correct table.

## Sheet\_number

This column denotes which sheet of the workbook to use. You can either use the index of the sheet (0 indexed) or the actual name of the sheet you would like to apply the mapping to.

## File Path

This is where the file is located relative to the top folder of the process. For instance, if your files are to be stored in Cap Leak/2022/Molina, this field would say 2022/Molina/

## Database Table

This is the name of the table you would like to write to in the tp-bisql-02.Finance database. You can put different tables for each file in the mapping.

## Header

This is the index of the header row for the sheet selected (0 indexed).

## Method

This is the method you would like to use to complete your SQL transaction. For this field you have 2 choices. Your first choice is to append the data to the table. To do so type append into the cell in the related column.

The other option is to truncate the table, which will remove all data from the table and replace it with the data in the workbook. To do this type truncate into the cell in the related column. The program will ask you to confirm that that is the operation you would like to undertake before moving forward with the SQL transaction.

## Import?

This field allows you to import or not import a certain file. This is a Boolean value (0 or 1). 1 to import, 0 to ignore. The program will print a statement for each file that is skipped.

## Archive Path

This is the path to which this file will be archived **after** mapping, but **before** forced datatyping. These paths are relative to [this](MCO%20Data%20Archives) base folder.

## Raw Archive Path

This is the location that the files will be copied to before any data manipulation. This is a full path to the location which you would like these to be archived in.

# Date Input

This is the tab that you will put the month and year that you would like to search for the data required. The month and year of the file must be separated by a hyphen. Nothing else matters in regards to the title, just that it is in the appropriate folder and the month is separated from the year by a hyphen.

Example:

Amerigroup Cap Leak 03 – 2022.xlsx

# Adding a Process

To add a process, create a new folder with file structure outlined [above](#_File_Structure). Then copy and paste a mapping file from another process and begin to work off that file.

# Adding a file to an existing process

To add a file to an existing process, all that needs to be done is to add a column to both the [[Mapping]](#_[Mapping]_tab:) tab of the mapping file and to the [[Meta]](#_File_Path) tab.

# Features to be added:

* Print process documentation feature
* Add a mapping type for non MCO processes that doesn’t rely on dates.
* ~~Add random key to tables so that mistakes can be quickly removed from the table.~~ 
  + Added file upload date instead auto generated by server April 2022
* Add a automated process to create a new procedure/mapping
* Enable multiple files with the same date for same MCO
* Enable raw table upload for cumulative roster upload like situations
* Add raw column list to compare to expected on mapping failure
* Implement cumulative roster