# City Council Triennial Redaction

# Overview

The code is designed to redact sensitive information in Excel reports used by the City Council. It works by masking values that are below certain thresholds, ensuring compliance with privacy and confidentiality requirements. The redaction is applied across various reports within the Excel file.

# Code Structure

The redaction process is managed by the script `redaction_triennial.py`, which relies on configurations specified in `redaction_config_triennial_04022024.py`. The main steps include:

1. **Copying Files** : The original Excel report is copied from a network location to a local directory.
2. **Applying Redaction** : The code iterates through each report tab in the Excel file, applying redaction rules based on predefined configurations.
3. **Saving the Redacted Report** : The redacted report is saved with the original content modified to hide sensitive information.

# Steps to Run the Redaction Code

## 1.Set Up Your Environment :

Ensure Python is installed on your computer.

Create folder: C:\Users\Ywang36\OneDrive - NYCDOE\Desktop\CityCouncil and C:\Users\Ywang36\OneDrive - NYCDOE\Desktop\CityCouncil\CCUnredacted to receive unredacted City Council triennial reports

Create folder R:\SEO Analytics\Reporting\City Council\City Council SY24\MM.DD.YY Triannual Report to receive final reports

Put Non-Redacted Annual Special Education Data Report_CW.xlsx in above folder, the 10 tabs will be append to it after run triennial generation code

Adjust eg. self.lastrow = 4042 as the correct number as report displays, delete the wrong tabs and regenated tabs, check if self.lastrow are the correct number.

Rename Non-Redacted Annual Special Education Data Report_CW.xlsx as Non-Redacted Annual Special Education Data Report_MMDDYYYY.xlsx and copy it to folder R:\SEO Analytics\Reporting\City Council\City Council SY24\MM.DD.YY Triannual Report

Install necessary Python libraries (e.g., `openpyxl` for Excel file manipulation). You can install these by running:

```
pip install openpyxl
```

## 2. Prepare the Files :

Ensure the original non-redacted report is saved in the specified network folder: R:\SEO Analytics\Reporting\City Council\City Council SY24\MM.DD.YY Triannual Report\

Ensure the `redaction_config_triennial_04022024.py` and `redaction_triennial.py` files are located in the same directory on your local machine.

## 3. Run the Script :

- Open a command prompt or terminal.
- Navigate to the directory containing the Python scripts.
- Run the following command

```
python redaction_triennial.py
```

## 4.Check the Redacted Report :

- After running the script, the redacted report will be saved in the following location on your desktop: C:\Users\[YourUsername]\OneDrive - NYCDOE\Desktop\
- Open the redacted Excel file and verify that the sensitive information has been appropriately masked.

# Code Breakdown

## 1. Importing Modules

- `openpyxl` : A library for reading and writing Excel files.
- `os` **and** `shutil` : Used for file operations like copying files.
- `TRIENNIAL_REPORTS_CONFIG_SY240402` : This is configuration imported from separate file that define redaction scopes and rules for different reports

# 2. Class Definition: `Solution`

The `Solution` class encapsulates all the functionality for the redaction process.

```
class Solution:
    def init(self, datestamp="06152024", date="06.15.24"):
        self.datestamp = datestamp
        self.date = date
        self.schoolyear = 'SY24'
        self.folderpath = fr'R:\\'
```

`__init__` **Method** : Initializes the class with default values for the date and file paths.* `datestamp` and `date` : These parameters represent the date format used in file naming.

- `folderpath` : This is the base directory where the original reports are stored.

**File Copying Method:** `copyonefile`

```
def copyonefile(self, src, dst):
shutil.copy(src, dst)
print('copying one file from {0} to {1} is complete'.format(src, dst))
```

- `copyonefile` **Method** : Copies a file from the source (`src`) to the destination (`dst`). This is used to copy the original unredacted report to a local directory before processing.
-

The `mask_excel_file` function is designed to apply a series of redaction and masking rules to specific worksheets within an Excel workbook. These rules are used to anonymize and protect sensitive data based on various conditions, particularly for Partial Services (PS) and Related Services (RS) reports.

# 3.Redaction Logic:

**1.Loading Workbooks and Worksheets :**

- The function starts by loading the main workbook (`filename`) and the unredacted workbook (`unredacted_filename`).
- It attempts to access the specified worksheet (`tab_name`) within both workbooks. If the worksheet does not exist, the function logs a warning and skips further processing.

**2.Column Masking :**

- If the `PS_flag` is set in the configurations, the function applies the `PS_column_masking` method, which masks values in specific columns for Partial Services.
- If the `RS_flag` is set, it applies the `RS_column_masking` method to mask values for Related Services, taking into account the specific type of report (e.g., by Superintendent, District, or School).

**3.Initial Masking :**

- The function calls the `initial_mask` method for each range specified in the configurations to apply initial redactions based on general masking rules.

**4.NA and Partial Encounter Redaction :**

- If the configuration includes `NA_Partcial_Encounter_Redaction`, the function applies specific redaction rules for cells marked with "N/A" or related to partial encounters using the `apply_na_redaction` method.

**5.Percentage Redactions :**

- The function applies redaction logic for percentage values in PS or RS reports by calling `apply_percentage_redaction_byPS` or similar methods depending on the flags set in the configuration.
- For specific cells marked with `100%`, it applies masking using `mask_smallest_numeric_and_percentage_byPS` or `mask_smallest_numeric_and_percentage_byRS`.

**6.Category and District Masking :**

- The function applies additional masking based on categories or districts:

- If `mask_by_category` is set and `PS_flag` is true, it uses `mask_by_samecategory_byPS`.
- If both `mask_by_category` and `mask_by_district` are set, it applies `mask_by_samecategoryanddistrict_byPS`.
- Similar logic is applied for RS flags using `mask_by_samecategory_byRS` or `mask_by_samecategoryanddistrict_byRS`.

**7.Final Masking Adjustments :**

- The function performs additional checks and adjustments to ensure that percentage cells are properly masked or unmasked as needed. This includes methods like `restore_percentage_cells_if_excessive_redaction`, `mask_0_percent_in_full`, and `mask_percent_having0_and_100_percent`.
- It also applies logic to handle potential over-redaction using the `overredaction_0` method.

**8.Unmasking for Specific Conditions :**

- If `RS_flag` is set, the function checks for specific conditions where numeric cells should be unmasked if they were over-redacted, particularly when paired with `0%` or `100%` percentages, using `unmask_numeric_having0_and_100_percent_non_bilingual`.

**9.Saving and Closing :**

- After applying all the masking and redaction logic, the function saves the modified workbook and closes it.