UHC Audit Automation

Technical Architecture Document

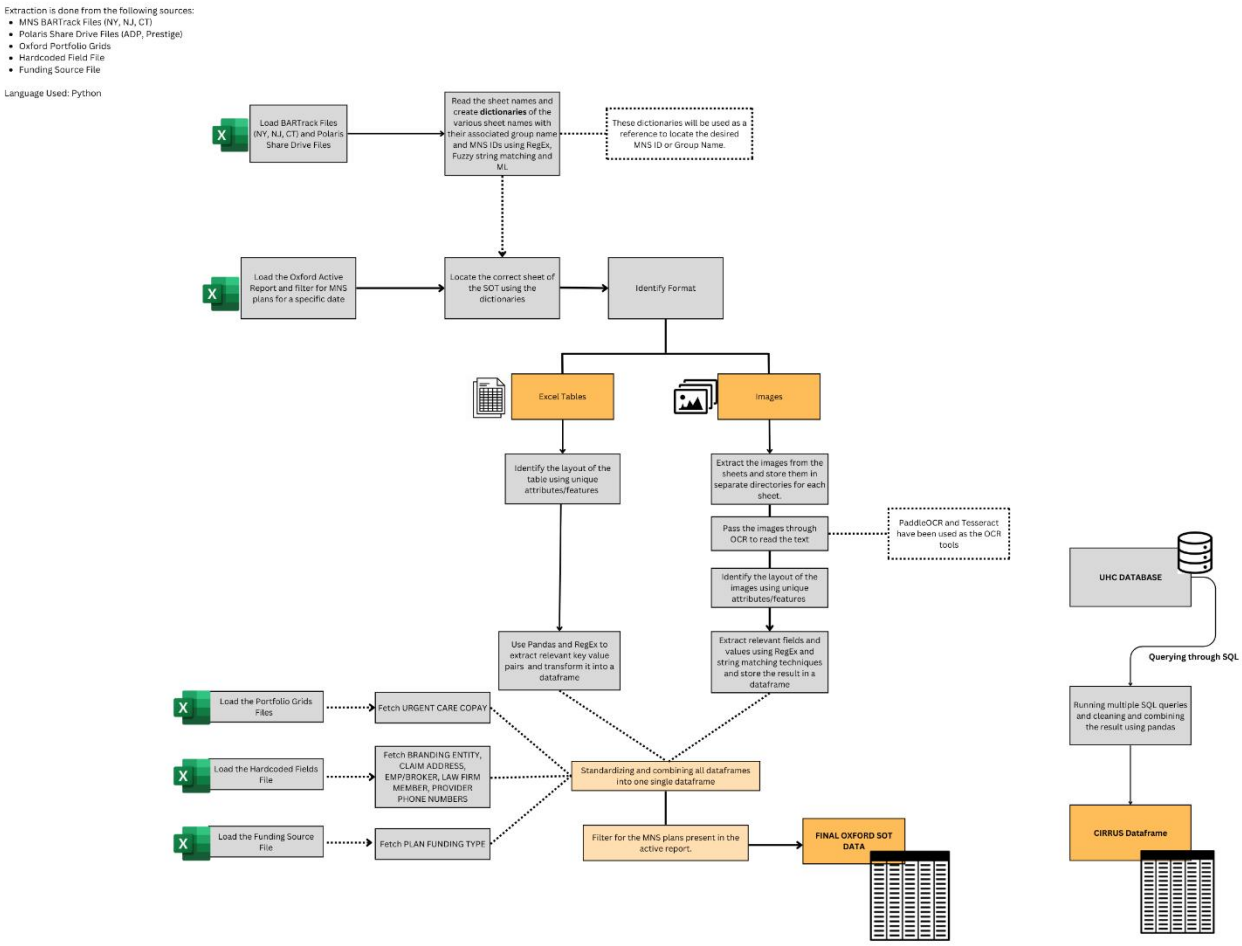
Version: 1

Date: 07/02/2024

Author: Subhadeep Choudhury

OXFORD EXTRACTION PROCESS

The below flowchart shows the overall architecture of the backend extraction process for Oxford Book of Business



Data Sources:

- MNS BARTrack Files (NY, NJ, CT)
- Polaris Share Drive Files (ADP, Prestige)
- Oxford Portfolio Grids
- Hardcoded Fields File
- Funding Source File

Process:

1. Loading necessary files:
   Load the Active Report and the BARTrack files for New York (NY), New Jersey (NJ), and Connecticut (CT). The pd.ExcelFile function is used to loop through each BARTrack file and extract and store the sheetnames in designated lists. In addition to BARTrack, portfolio grid files, hardcoded fields file and funding source file is also loaded.
2. Creating Dictionaries:

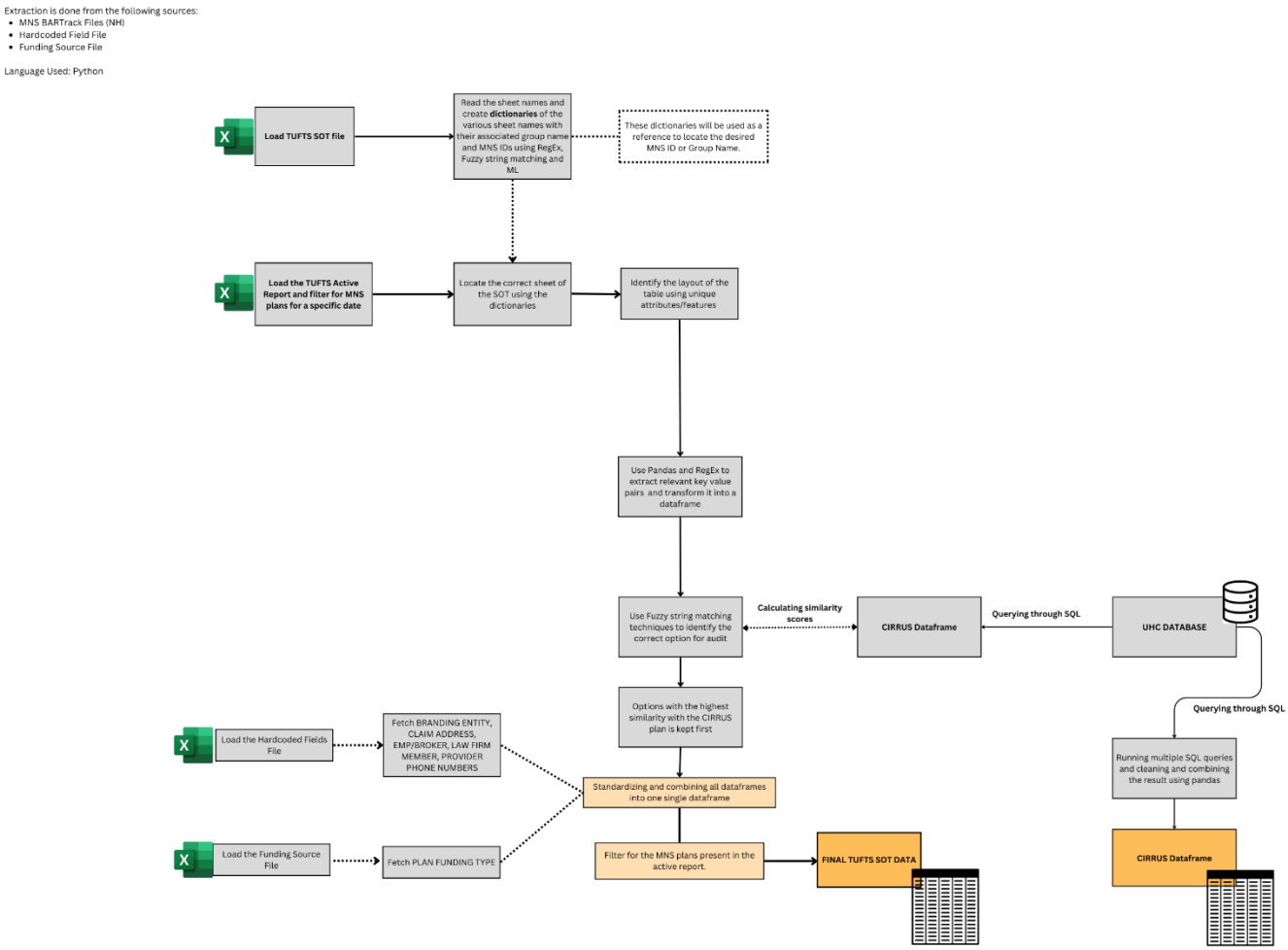
Dictionaries are created that map sheet names to their associated group names and MNS IDs using techniques like Regular Expressions (RegEx), fuzzy string matching, and machine learning. These dictionaries serve as references to locate the desired MNS ID or Group Name in subsequent steps.

3. Filtering the Active Report:
   The active report is filtered for only MNS plans based on the user input (date) and the list of plan IDs and group names are generated.

4. Locating Correct Sheets:
   Use the dictionaries created in the previous step to locate the correct sheet of the Source of Truth (SOT).
5. Layout Identification:
   i. For excel tables, identify the layout of the Excel tables using unique attributes or features present in the data and apply correct pandas and regex based functions custom made for that specific layout type to extract the relevant data.
   ii. For images, first extract images from the identified sheets. These images are then stored separate directories based on their respective sheets. The extracted images are passed through OCR tools such as PaddleOCR and Tesseract to read the text contained within the images. Similar to Excel tables, the layout of the images is identified using unique attributes or features.Finally, RegEx and string matching technique based functions are used to extract relevant fields and values from the images for each layout type.
6. For the relevant plans, UC Copay is fetched from portfolio grids file
7. For the relevant plans, 6 hardcoded fields fetched from the hardcoded field file
8. The resultant dataframes are cleaned, standardized and combined into a uniform dataframe which is the SOT dataframe.
9. The multiple SQL queries are run and the outputs are cleaned, standardized and combined into a single dataframe for the relevant plans using mysql.connect, pandas and regex. The result gives us the CIRRUS dataframe.

TUFTS EXTRACTION PROCESS

The below flowchart shows the overall architecture of the backend extraction process for TUFTS Book of Business



Data Sources:

- TUFTS SOT File (NH)
- Hardcoded Fields File
- Funding Source File

Process:

1. Loading necessary files:
   Load the Active Report and the SOT file. The pd.ExcelFile function is used to loop through file and extract and store the sheetnames in designated listsCreating Dictionaries:
   Dictionaries are created that map sheet names to their associated group names and MNS IDs using techniques like Regular Expressions (RegEx), fuzzy string matching, and machine learning. These dictionaries serve as references to locate the desired MNS ID or Group Name in subsequent steps.

2. Filtering the Active Report:
   The active report is filtered for only MNS plans based on the user input (date) and the list of plan IDs and group names are generated.

3. Locating Correct Sheets:
   Use the dictionaries created in the previous step to locate the correct sheet of the Source of Truth (SOT).
4. Layout Identification:
   The type of layout of the Excel tables is identified using unique attributes or features present in the data and apply correct pandas and regex based functions custom made for that specific layout type to extract the relevant data.

5. The resultant dataframes are cleaned, standardized and combined into a uniform dataframe which is the SOT dataframe.

6. The multiple SQL queries are run and the outputs are cleaned, standardized and combined into a single dataframe for the relevant plans using mysql.connect, pandas and regex. The result gives us the CIRRUS dataframe.

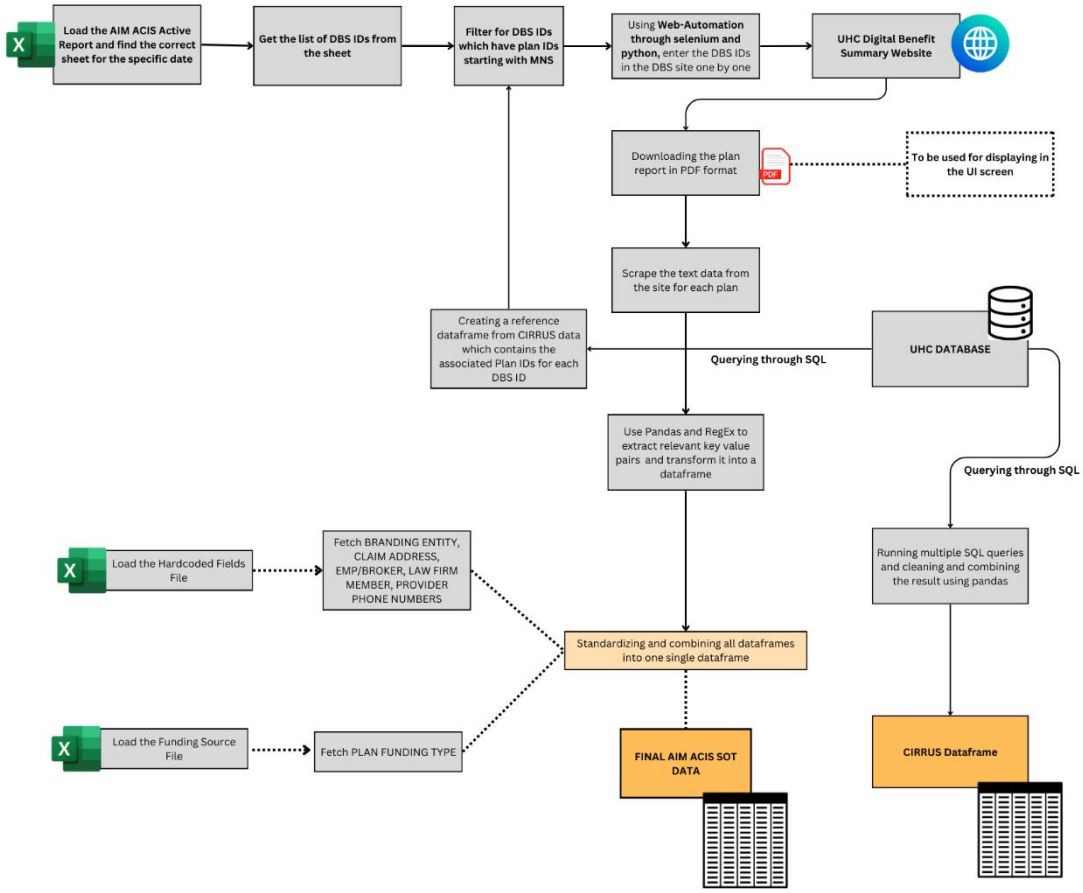
7. Calculating Similarity Scores: Each plan in SOT may have multiple options, but only one of the option is present in CIRRUS. Hence a fuzzy string matching function is used to calculate similarity between each option and the CIRRUS plan, and the plan with the highest score is shown first to the user.

AIM ACIS EXTRACTION

The below flowchart shows the overall architecture of the backend extraction process for AIM ACIS Book of Business

Extraction is done from the following sources:
- UHC Digital Benefit Summary Website
- Hardcoded Field File
- Funding Source File

Language Used: Python



Data Sources:

- UHC Digital Benefit Summary Website
- Hardcoded Fields File
- Funding Source File

1. Loading necessary files:

   The active report, hardcoded field file and the funding source files are loaded.

2. Fetching DBS IDs:

Based on the date input from the user the active report is searched for the sheet containing the plans for the given date. From the sheet the DBS IDs are pulled and stored as a list.

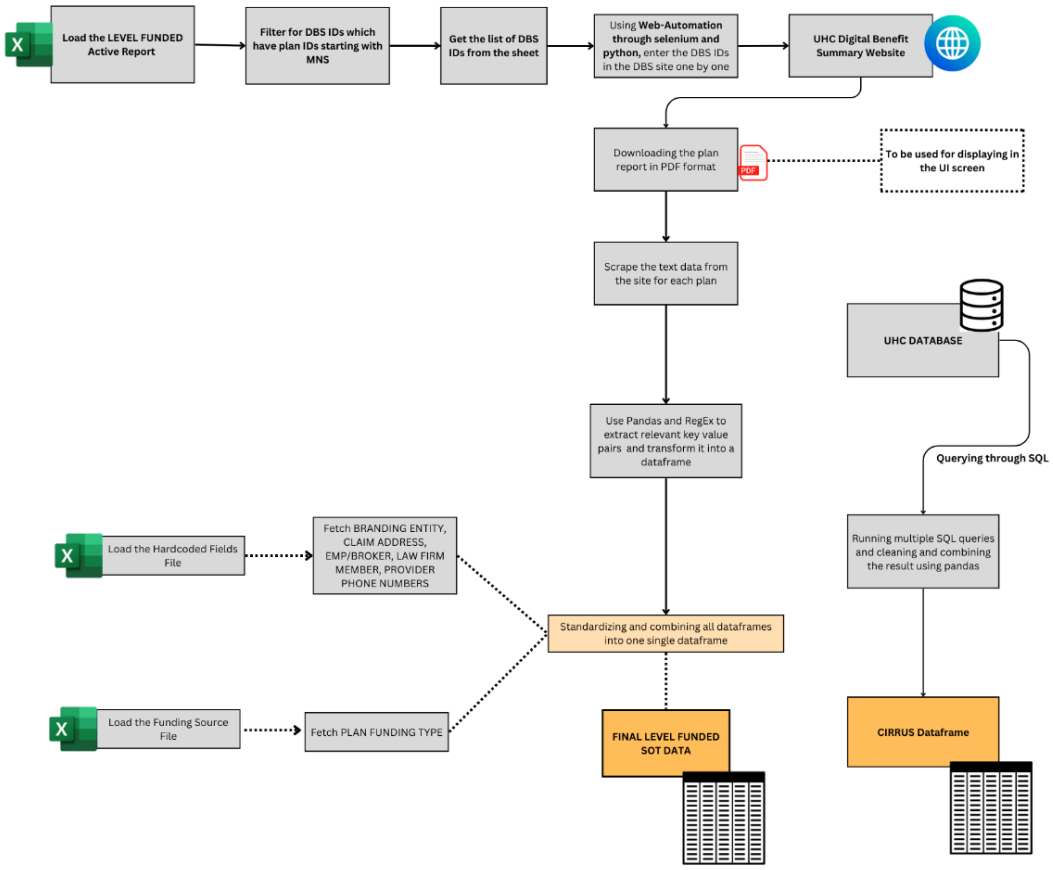
3. Generate a reference dataframe from CIRRUS: From CIRRUS, a dataframe containing the PLAN ID and the DBS ID is generated by running SQL queries, cleaning, and combining the results using Pandas. This is done to find the associated MNS IDs for each DBS ID, as the active report does not contain PLAN ID, but contains only DBS IDs.
4. Filter DBS IDs which are associated with Non-Standard Plans: Using the reference table we filter the only those DBS IDs which have plan IDs starting with MNS. DBS IDs that do not have entries in CIRRUS at all and those with standard (M0) plan IDs are omitted.
5. Web-Automation to UHC Digital Benefit Summary Website:
   Use web automation through Selenium and Python to enter the final list of DBS IDs in the UHC Digital Benefit Summary Website one by one.
6. Download plan report in PDF and scrape text data: The plan report in PDF format are downloaded for each plan from the website and web-scraping is done to fetch the text information. The downloaded PDFs will be used to represent the SOT on the UI screen.
7. Cleaning the data and transforming it into dataframes: Relevant key value pair information is then extracted from the scraped text for each plan using Pandas and Regex. The results are stored in dataframes.
8. Load the Hardcoded Fields File:
   Load the hardcoded fields file and fetch details like BRANDING ENTITY, CLAIM ADDRESS, EMPLOYER ID, LAW FIRM MEMBER, PROVIDER PHONE NUMBERS.
9. Load the Funding Source File:
   Load the funding source file and fetch PLAN FUNDING TYPE.
10. Standardizing and Combining the Dataframes: The Dataframes are then merged with the hardcoded fields and funding type and then standardized and combined into one final dataframe which is the SOT Dataframe that would be displayed in the UI.
11. The multiple SQL queries are run and the outputs are cleaned, standardized and combined into a single dataframe for the relevant plans using mysql.connect, pandas and regex. The result gives us the CIRRUS dataframe.

LEVEL FUNDED EXTRACTION PROCESS

The below flowchart shows the overall architecture of the backend extraction process for LF Book of Business

Extraction is done from the following sources:
- UHC Digital Benefit Summary Website
- Hardcoded Field File
- Funding Source File

Language Used: Python



Data Sources:

- UHC Digital Benefit Summary Website
- Hardcoded Fields File
- Funding Source File

1. Loading necessary files:

   The active report, hardcoded field file and the funding source files are loaded.

2. Fetching DBS IDs:

Based on the date input from the user the active report is filtered for MNS plans for the given date. The DBS IDs are then stored in a list

3. Web-Automation to UHC Digital Benefit Summary Website:
   Use web automation through Selenium and Python to enter the final list of DBS IDs in the UHC Digital Benefit Summary Website one by one.
4. Download plan report in PDF and scrape text data: The plan report in PDF format are downloaded for each plan from the website and web-scraping is done to fetch the text information. The downloaded PDFs will be used to represent the SOT on the UI screen.
5. Cleaning the data and transforming it into dataframes: Relevant key value pair information is then extracted from the scraped text for each plan using Pandas and Regex. The results are stored in dataframes.
6. Load the Hardcoded Fields File:
   Load the hardcoded fields file and fetch details like BRANDING ENTITY, CLAIM ADDRESS, EMPLOYER ID, LAW FIRM MEMBER, PROVIDER PHONE NUMBERS.
7. Load the Funding Source File:
   Load the funding source file and fetch PLAN FUNDING TYPE.
8. Standardizing and Combining the Dataframes: The Dataframes are then merged with the hardcoded fields and funding type and then standardized and combined into one final dataframe which is the SOT Dataframe that would be displayed in the UI.
9. The multiple SQL queries are run and the outputs are cleaned, standardized and combined into a single dataframe for the relevant plans using mysql.connect, pandas and regex. The result gives us the CIRRUS dataframe.