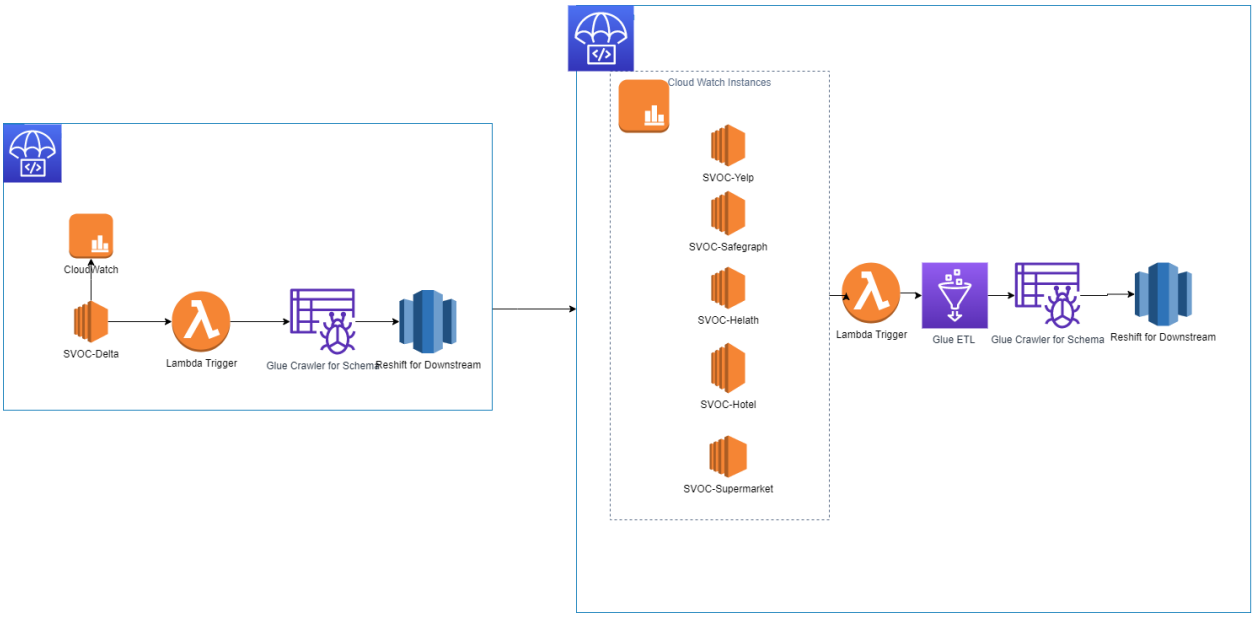
Capstone Project 2 Milestone Report 2

Flow Diagram for Third-party Enrichment Process:



The above flow has the following steps:

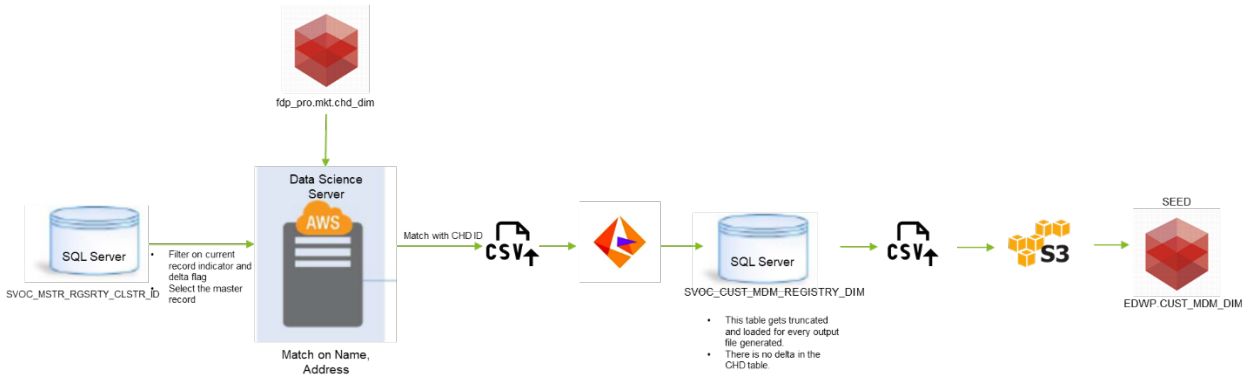
1. Sourcing & cleansing of USBL, SSMG, and Fresh Point data for logically active customers (which were either created or had a transaction in last 12 months). The cleansing rules have been attached in the cleansing section.
2. Handling delta for the following 4 scenarios:
    a. New customer
    b. Ownership change (Change in name and not in address)
    c. Location change (Change in address and not in name)
    d. Logically Inactive records
3. Perform in file deduplication clustering for all the records in the first load and then for the delta records in the subsequent loads based on Name and Address fields
4. Load the output to a staging table and identify the master record for each cluster following the below rule:
    a. Sort based on source in the following order USBL, Fresh Point, SSMG
    b. Sort on customer names within each source
    c. Sort on concat Id within each sorted name in a source
5. Use the master record of each cluster to perform cross match with all third party data, including safegraph, CHD, and Yelp. To accelerate the calculation, each match is run on an individual EC2 instance.
6. Populate the relevant third party enrichment data for the respective cluster IDs after the CHD match
7. Load the SQL and SEED table with the final data.

Data Cleansing Rule

Description	Rule
Number of OPCOS for the source	Distinct count of the OpCo's
Number of OPCOS we have data available for	Total OpCo's – distinct count of OpCo's
Total number of unique customers for the available OPCOS	Distinct count of the customer and OpCo
Total Number of Logically Active Customers	Distinct count of the customer and OpCo for a customer who has had a sales transaction with Sysco in the last 12
Total Number of Invalid Address = a+b+c+d+e Total Valid Address = Total Address – Invalid Count	1. Replace all the special characters with an empty string (gsub("[]\\?!\"\'#$%&(){}+*/:;,._`|~\\[<=>@\\^-]", "", x) and convert everything to uppercase. 2. Count the number of records which have an empty or null address_1, address_2, state, city, zip and remove these records before the next step **[Count: a]** 3. Count the number of address_1, address_1 fields which are empty or null and remove these records before the next step **[Count: b]** 4. Count the number of records which have the state, city, and zip empty but have an address in the address fields and remove these records before the next step. **[Count: c]** 5. Count the number of records for which the address fields have only numbers and remove these records before the next step. **[Count: d]** 6. Count the records which have the following strings in them: ('CLOSED','INACTIVE','NOT ACTIVE','DO NOT','DONT USE', 'DONT USE', 'BAD DEBT', 'BAD DEPT', 'RESETUP OF ACCT', 'NOT VALID', 'INVALID', 'GROWTH PROGR', 'WRITE OFF', 'NEED PHONE','TEST', 'TEST ', 'PLACEHOLDER', 'UNKNOWN', 'DISCONTINUED', 'DUPLICATED', 'USE OTHER', 'EMPLOYEE', 'RETURN TO', 'SYSCO') **[Count: e]**
Total Number of Invalid Names = a+b+c+d Total Valid Names = Total Names – Invalid Count	1. Replace all the special characters with an empty string (gsub("[]\\?!\"\'#$%&(){}+*/:;,._`|~\\[<=>@\\^-]", "", x) and convert everything to uppercase. 2. Count the number of records which have an empty or null address_ and remove these records before the next step **[Count: a]** 3. Count the number of records for which the names are only numbers and remove these records before the next step. **[Count: b]** 4. Count the records which have the following strings in them: ('CLOSED','INACTIVE','NOT ACTIVE','DO NOT','DONT USE', 'DONT USE', 'BAD DEBT', 'BAD DEPT', 'RESETUP OF ACCT', 'NOT VALID', 'INVALID', 'GROWTH PROGR', 'WRITE OFF', 'NEED PHONE','TEST', 'TEST ', 'PLACEHOLDER', 'UNKNOWN',

	'DISCONTINUED', 'DUPLICATED', 'USE OTHER', 'EMPLOYEE', 'RETURN TO', 'SYSCO') **[Count: c]** 5. For **SSMG**, records which have an acc_typ_cd = EMPLOYEE and cust_id like 'e%' **[Count: d]**
Total Number of Valid Records for Match	Count the number of records after the intersection of all the records which have both valid names and valid address.
Logically Active	The customers who have had a transcation with Sysco in the past 12 months.

Third party data Matching & Master Registry



Once the view is populated along with the Record Rank from the SVOC process, we trigger the third party matching algorithm with a scheduled AWS lambda function. To achieve parallel computing, the same matching data science model is loaded on 4 individual EC2 instances. Each individual EC2 instance will ingest data from different databases (Safegraph, Yelp, CHD and Retail store). Once the four third party match are completed, the output file is placed in a temporary folder in server. This location is mounted on Informatica Server and we use the file as a source for our ETL process to generate Master Registry table in SQL Server.

Matching with Python Algorithm:

The matchingMainCHD.py uses Record Linkage (BigDataLinkage) method from RecordLingake library to identify if the master record from the view matches with the CHD data provided by a 3rd party within each state. Within each state and city, it searches for similar records based on the string match for customer name, customer street address, customer segmentation and phone number. The algorithm matches either the internal data or third party data with an appropriate third party record based on the threshold value in the configuration file. For all the records which did not get populated with a match data, the match columns are populated as NULL. The match data is used to enhance the internal data for customer insights. There are different user cases for the matching output, e.g. lead to

prospect potential, customer basic metric, etc. The final output of the algorithm is a csv file (SVOC_MSTR_RGSRTY_CLSTR_ID_CHD.csv.) in the ADM server.

To improve the accuracy of the original fuzzy-match based algorithm, I implemented Supervised Machine Learning Model instead of Fuzzy Matching: Consistent measures of matching probability. After the unsupervised labeling process as described in the first report, human supervised data label was performed. This process removed obvious mistakes of data records mismatch. After this ground-truth label process, different data science models are applied to determine the best model for this dataset. As shown in Figure 1, ensemble method, XGBoost achieved the best AUC accuracy among other methods, and therefore is chosen as the final model.

The new data science model also contains extensive Feature Engineering for Attributes-based matching: Comprehensive consideration of various types of evidence for matching. Besides the original attributes, customer name, address, other features are included in the final DS model. These features, e.g. customer segmentation and phone number are also added. From the feature selection chart shown in Figure 2, these two features also shown great importance in the final model.

To effectively accelerate data model development speed, the third party library is deployed for the index-pair generation between Internal and External data: Solved the memory and computation efforts of Fuzzy Match. It is worth to notice that only the index-pair part of the record linkage library is used in this data science model. Other parts of the unsupervised classifiers from RecordLinkage library are not used. The reason for this is that the unsupervised classifier cannot establish a rigorous threshold, and will result in poor record matching results.

Finally, there are other possible products on the market, however such products also require many rounds of data labeling, and will result in high cost for building a customaries machine learning model. In house software that can be run economically on a serverless virtual machine: AWS FindMatches is expensive and use the similar algorithm.

A detailed comparison of the original unsupervised ML algorithm and supervised ML algorithm is shown in Figure 3. It can be seen that the matching percentage has been improved by 7% after implemented the supervised ML algorithm.

Different machine learning models are applied to the data, and their AUC scores are used to benchmark their performances:

	Method	Best_Score
0	XGBoost	0.996654
4	SGDClassifier	0.996460
5	ExtraTreesClassifier	0.996448
6	SVC	0.996434
2	LogisticRegression	0.996395
3	AdaBoostClassifier	0.996209
7	KNeighborsClassifier	0.992302
1	RandomForest	0.980872

Figure 1: Comparison of different DS models.

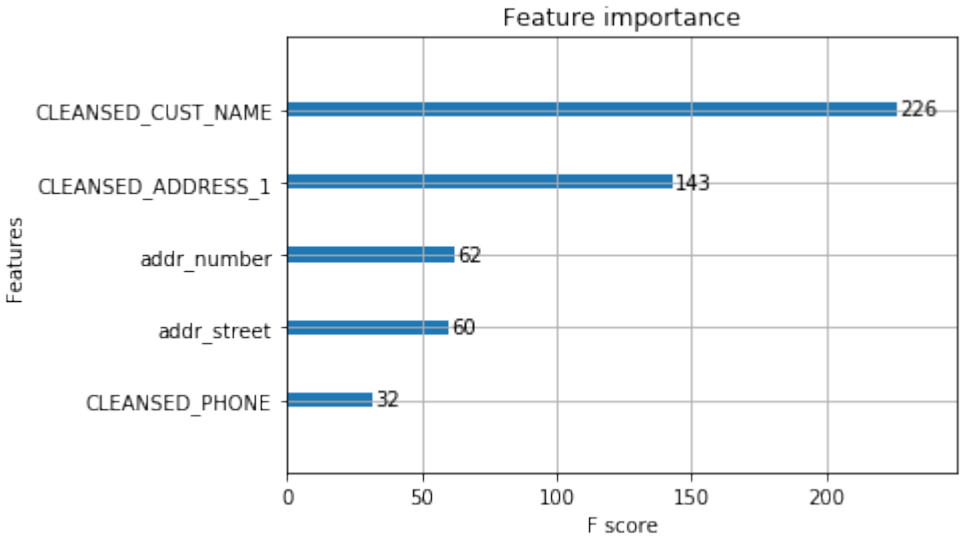


Figure 2: Feature importance comparison.

	ML Algorithm	Unsupervised Algorithm	Difference
Total Matched Count	242011	223747	18264
Total Count Pct for Sysco Data	70%	64%	6%

Figure 3: A comparison of Supervised and Unsupervised ML algorithms.

Moving data to SEED

Linux based shell script that uses the 'AWS CONFIGURE' command to connect to the S3 bucket where the SVOC file (REDSHIFT_SVOC_MSTR_RGSRTY_CLSTR_ID_CHD.txt) will be placed. Once the file has been moved from the INFA source location (/mnt/csv2/), it will then be moved to the S3 bucket (//OpCo Data/MDM/). Upon landing to the S3 bucket, the Linux script will connect to the Redshift server using the SVC_MDM account and truncate the existing CUST_MDM_DIM table. Once the table has been truncated, a AWS COPY command will be executed loading the file from S3:

copy edwp.cust_mdm_dim_bkp (mdm_src_sys_cd, cust_concat_id, co_skey, cust_skey, co_nbr, cust_nbr, cust_shipto_nbr, src_cust_nm, src_addr_line_1_txt, src_addr_line_2_txt, src_addr_line_3_txt, src_cty_nm, src_stt_nm, src_zip_cd, src_cntry_nm, sysco_phone, cln_cust_nm, cln_addr_line_1_txt, cln_addr_line_2_txt, cln_addr_line_3_txt, cln_cty_nm, cln_stt_cd, cln_zip_cd, cln_cntry_cd, mdm_cust_clstr_id, clstr_or_isl_ind, clstr_typ_cd,clstr_src_cnt, clstr_lbl_nm, clstr_cnfdnc_scor_rng, chd_id, chain_id, location_pid, owner_pid, chd_name, chd_address, chd_city, chd_state, chd_zip, chd_menu_type, chd_operation_type, chd_market_sgmnt, chd_customer_type, src_sys_cd, mstr_rec_rank) from 's3:// /OpcoData/MDM/REDSHIFT_SVOC_MSTR_RGSRTY_CLSTR_ID_CHD.txt'

Marketing Analytics

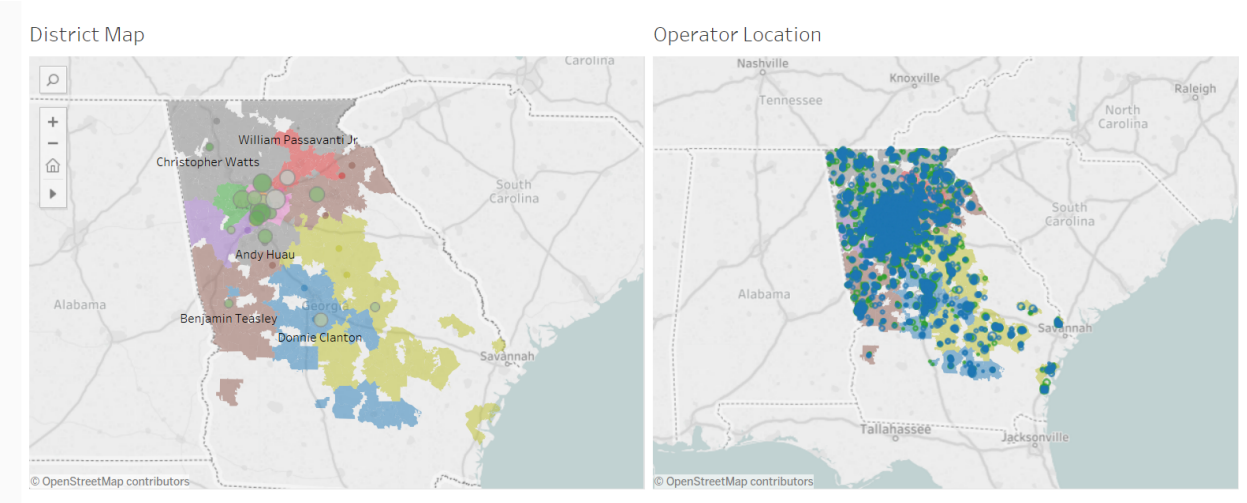
The Marketing Analytics Team is the consumer of the data in SEED. Marketing Analytics uses this data to identify the market potential of the customers Sysco is dealing with and identifies the market potential of the customer Sysco needs to engage in a business with.

The inputs to the marketing analytics team are the customer master data, which has the clustered data across the 3 lines of business, and sales transactional data, which helps identify the market potential for business.

SQL Query used for aggregating the sales data:

MAQuery.txt

Screenshot of the Marketing Analytics Dashboard:

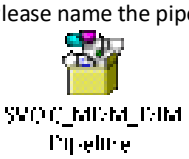


AWS Data Pipeline and Lambda

BI SEED team requires that all processes be schedule via a AWS Lambda using CRON job to kick off the process every week at 0100 hrs. on Thursday nights. The AWS Lambda is written in Python 2.7 and looks to identify the SVOC Data Pipeline and the status of the Data Pipeline and whether any previous Data Pipelines are still running. If the previous SVOC pipeline has finished and the Pre-Day close has finished, the Lambda will call the SVOC_Pipeline and write a record to the Aurora Tracking table that the pipeline has started

The SVOC data pipeline once started via the Lambda, calls the workflow () and waits for the workflow to complete. Once the workflow is complete, the data pipeline will update the Aurora Tracking table is complete. If there is an error in completing the workflow, a log with update showing the workflow and pipeline failed.

1. Upload JSON file for Pipeline
    a. Please name the pipeline “SVOC_MDM_DIM”



    b.
2. Upload Lambda Zip File
    a. Please name the pipeline “SVOC_MDM_DIM”

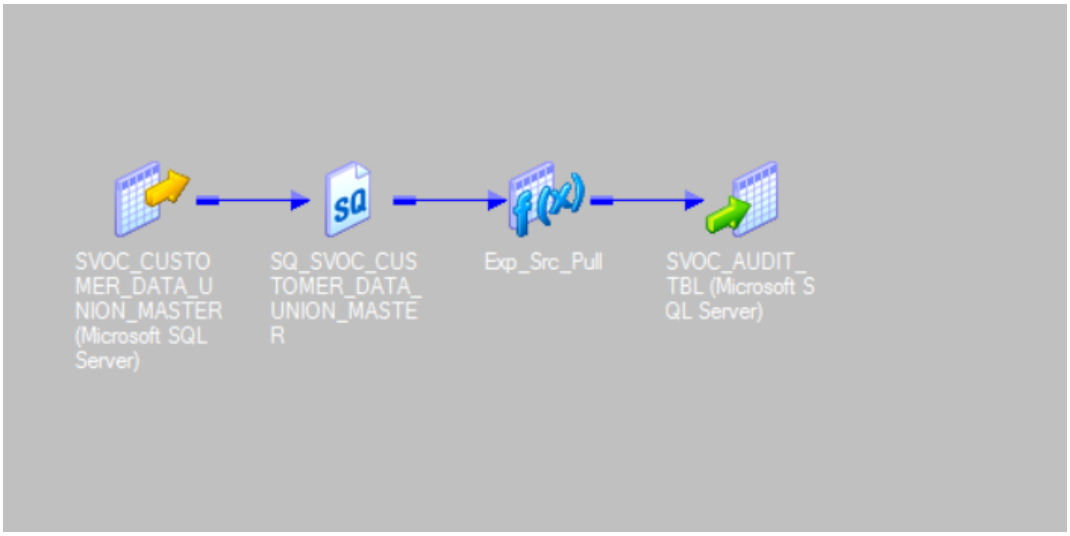
b. 

3. Create CloudWatch Event
    a. Add CRON job for Bi-weekly at 8:00 AM CST (13:00 GST)
    b. 0 0 13 * * WED

SVOC Process Statistics Report

A mapping has been created to pull all the statistics and load in the audit table, details below. This process will run at the end of SVOC workflow and populate the statistics in the Audit Table. A date field has been added to enable any trend analysis in future.

Mapping Screenshot:



Screenshot of the output in Audit Table: