Data Engineering: Capstone Project

E-Rate Data Import for App Back-End

Introduction

Broadband is a wireless access point distributor and installer. Broadband is taking part in a discount telecommunication services program managed by the Universal Service Administrative Company (USAC). Within the program educational institutions can apply for government funded subsidies for telecommunication products and services.

The Goal

A data pipeline has been created in Airflow to import the E-Rate Form 470 details, along with the itemized service requests outlined in the form. Once imported, an estimated appraisal value is generated, and the sales opportunity is assigned to a sales representative.

This document will outline the process taken to build the data model and its corresponding pipeline.

Defining the End Use Case

The data model is designed to support the back-end of an app. As such, the data model does not resemble a data warehouse. For instance, the basic information retrieved from the 470 form is then distributed over 3 Postgres tables to meet 2nd Normal Form.

Being an app back-end data model, the database is susceptible to heavy writes and complicated read operations (i.e. joining of multiple tables to present related data). This data model works for the sales app because we want to reduce duplicated information, taking action on current, relevant data.

Meeting Project Criteria

Based on the project outline there are three stipulations on the dataset:

- At least two data sources
- More than 1 million lines of data
- At least two data sources/formats

Data Sources

To complete the pipeline, two datasets from USAC's Open Data API are imported. To meet the requirement of a "second data source" Postgres tables containing "regional sales assignments" have also been created and added to the Postgres database.

Dataset Size (> 1M lines of Data)

USAC provides, within their Open Data splash pages, the running number of rows within a given dataset. These data sets go back to 2016, when they implemented this JSON rendering HTTP API. Prior to 2016

USAC utilized an HTTP API that rendered the data in CSV or XML format. As of November 8, 2021 USAC's HTTP API contains:

- 192,129 rows in the E-Rate Open Competitive Bidding: Basic Information dataset
- 1,195,594 rows in the E-Rate Open Competitive Bidding: Services Requested dataset

Data Formats

- Both of USAC's HTTP API endpoints provides data in a JSON format
- Postgres tables containing sales rep assignment logic provide a SQL/Python Pandas data format
- The products and services pricing guide was moved to an XML document, stored in AWS S3 bucket, to expand the data format diversity

Exploring the Data Sources

Company Data

As mentioned, Broadband (the company) has a database consisting of sales representatives; the regions (i.e. states within the USA) they are responsible for and monetary deal value of which the sales opportunities would be distributed.

Any data quality issues found within this system will come human error on the data clerks' part. However, the sales app would be expected to mitigate the likelihood of these issues. For example, no two sales representatives can be assigned to the same region and deal value. Since the data pipeline is not importing this data from an outside source, it assumes that this data is accurate.

Pricing Guide

Our Airflow pipeline utilizes an XML file stored in an AWS S3 bucket. This file is also considered an internal, corporate data source and is not validated within the data pipeline.

"E-Rate Open Competitive Bidding: Basic Information" Dataset

API Link: https://opendata.usac.org/resource/jp7a-89nd.json

The initial source of data for the pipeline is the basic information for USAC's E-Rate Form 470. Being data from a third-party, data quality issues is highly anticipated. The following table outlines the issues encountered during the data import process and how the issue was mitigated:

Issue	Example	Mitigation
Revisions – applicants can submit Form 470 multiple times	form_version field can be *Original* or *Current*	• Sort submissions by last_modified field; then process the latest form version • Use of Pandas' *drop_duplicates* function to process the latest form version • On primary key conflict Postgres will use the UPDATE

		command instead of INSERT
Missing data	statewide_state field may be left out of API json rendering	Use of Pandas' *reindex* function to add null fields to row when absent
NaN values – issue appears at SQL INSERT execution	Longitude and Latitude fields are imported as NaN	Use of Pandas' *replace* function to change NaN values into None

"E-Rate Open Competitive Bidding: Services Requested" Dataset

API Link: https://opendata.usac.org/resource/39tn-hjzv.json

This dataset is an expansion, and itemization, of the *category_one_description* and *category_two_description* fields from the "Basic Information" dataset. The data quality concerns here reflect those outlined in the table above, with one additional consideration:

Issue	Example	Mitigation
Multiple columns for similar information	• function field may contain the value of *Other*; while other_function field contains the desirable value • manufacturer field may contain the value of *Other*; while other_manufacturer field contains the desirable value	When other_function field contains a value concatenate the field values

Technologies Used

Airflow

The pipeline is not an overly complex one. However, simply running a Python script would not be sufficient. Airflow has been selected to manage the automation for its modular approach to designing and monitoring the pipeline. Another component to the sales process is monitoring when educational institutions have selected a vendor for their services and apply to USAC for their discounts. With Airflow, I can easily create a second DAG to independently import USAC's E-Rate Form 471 data, or I can add new operators to the existing DAG without changing code to the existing operators. If added to the existing DAG, changing the DAG dependencies is less invasive than, for example, changing Python code in a Spark script.

Postgres Database

Postgres has been selected as the underlining database for the app. Although any relational database could have been selected, for the purpose of this course an AWS supported database seemed more appropriate. During the development of this pipeline a local instance of Postgres was used. However, in moving to production, an AWS RDS instance can also be stood up and used.

With the increased possibility that E-Rate Form 470s can be resubmitted the same day, or over time, having Postgres' *ON CONFLICT* feature also makes it ideal for dealing with updates. In the project, there is also a case for which “ON CONFLICT” I chose to *DO NOTHING* since the data is being considered immutable.

Why not Cassandra or Redshift

In taking a moment to contrast the question “why Postgres” with the questions “why <u>not</u> Redshift” and “why <u>not</u> Cassandra”.

- Cassandra would not be an optimal database because it is very much inclined to analytical workloads. This project required a database that would use JOINS and ad-hoc queries.
- Redshift would fair better than Cassandra due to its RDS nature. However, being a columnar storage rendition of Postgres, it may underperform during heavy row processing activities like pricing a service request or processing individual applications.

AWS S3 Buckets

S3 was actually selected out of convenience. During the designing of this pipeline the pricing XML file was to be stored on the Airflow server. However, challenges providing an appropriate relative file path seemed ridiculous to combat with. Instead, it made more sense to use an external fileshare for both the auxiliary files (i.e. the pricing guide) and the log file storage.

Process Steps

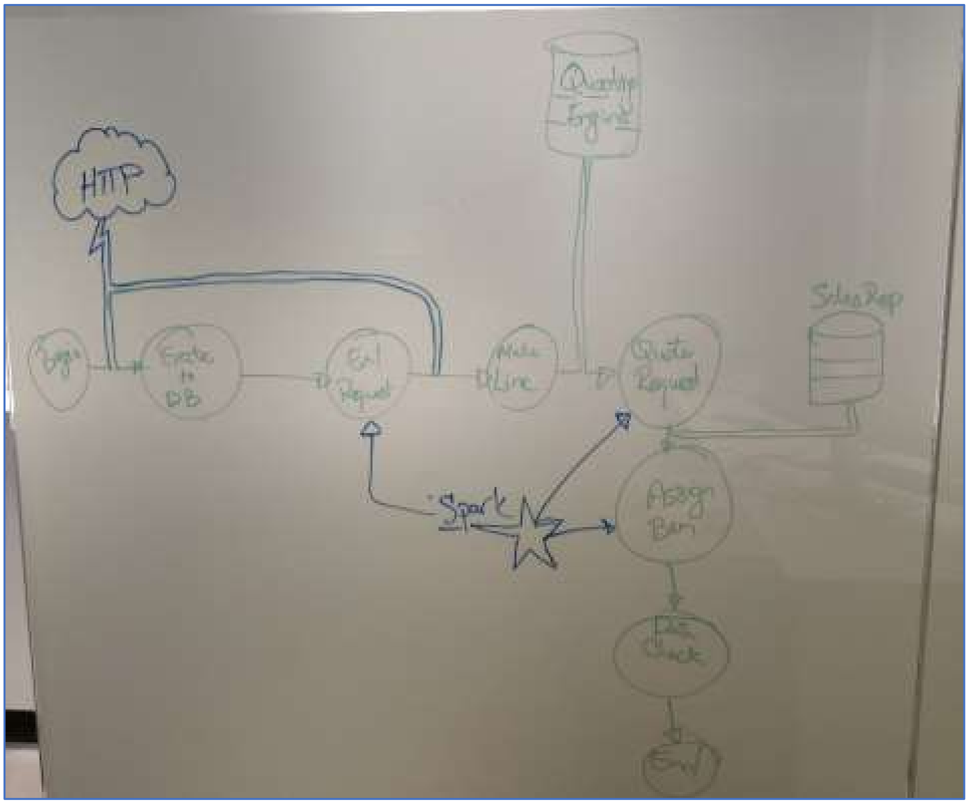

Figure 1:Sketched Pipeline

The process taken during this pipeline is a rather simple one. It consists of 2 HTTP API calls; 2 natural language processes and one classification process.

1. Import E-Rate Form 470 Basic Information into Postgres Database

The pipeline begins with making a API call. USAC's API has been built on the Socrata platform; which allows for very flexible queries to the data set. Using the query function, data is requested based on the *last modified date* of the form. Once received, the records are organized chronologically and preceding duplicates are dropped. The fields are then extracted, arranged to match the form; billing entity and requester contact entities and loaded into the Postgres database.

2. Mark Form 470 Records as Service Requests Relevant to our Business

Run a query to retrieve any form records where the applicable request flag is not set. The *category two description* field is then reviewed to determine if the form has requests that align with Broadband's provided services. The records are then updated with the boolean value generated by the NLP function.

3. Import E-Rate Form 470 Service Requests into Postgres Database

This is the second call made to a USAC API. This request query differs in that it is not queried by last modified date but repeats the API call on the application number. This increases the run-time for this step but helps reduce excessive data storage and processing in steps downstream. The retrieved data is then inserted into the Postgres database.

4. Evaluate Service Request Monetary Value

A query is run to retrieve service requests with the appropriate function but has not been flagged as a relevant request. A caveat is that the application requires a request where Wireless Access Point(s) or Wireless Controllers are being requested for purchase and/or lease.

Each record is matched to a node in the XML pricing guide. The pricing node and the service request record is then passed into another function to determine the monetary value. Each service function has its own data inconsistencies, so we attempted to correct unconventional requests.

Lastly, the monetary value is recorded to the service request record, and a running total is inserted into the form record.

5. Assign E-Rate Form 470 to a Sales Representative

Now that the applications have a monetary value, the application can be assigned to a Broadband sales representative. Applications are assigned by their monetary value and the state for which the requester resides. The relevant, unassigned applications are queried from the Postgres database and examined to determine which sales representative it will be assigned to.

6. Data Quality Checks

Lastly, a .json file is created to capture significant statistics of the import. An atomic-level query is run to get the details about the import behavior. Additional queries are then run against the retrieved data to capture the desired statistics. The statistics are then placed into the report templated and saved to an AWS S3 bucket.

Scheduling

I would propose that this pipeline is run once a day. The number of daily records is low and can be imported once a day. One thing to consider is that the import will contain Form 470 records from the previous day, not the execution day.

Data Quality Checks

During the quality check a variety of quality checks are completed. The two I chose to highlight are:

1) The false positive statistic highlights applications that contained keywords within the *category_two_description* field but did not actually meet the more thorough examination during *EvaluateRequestsOperator*. This provides import credibility because it allows analyst to refine the Natural Language Processing component in *MarkBidRequestsOperator*.
2) There is a catchall Sales Rep account (ID: 00-000-0000). This provides import credibility by catching applicable applications that land outside of the assignment matrix defined by the sales_state_*assignment* table.

Hypotheticals

What if the data was increased by 100x?

If the data was increased I would change two scheduling setting and one technology setting. The two scheduling changes would be:

1) switching the DAG schedule from daily to hourly. This would also require changes to *BidBasicInfoToPostgresOperator*, *AssignSalesRepOperator* and *UsacQualityCheckOperator*, so that the API and SQL queries review the date fields on an hourly basis.
2) switching the queries from a day before the execution date, to the same day as the execution date.

The technology change I would make would be to utilize Spark on *EvaluateRequestsOperator*, *AssignSalesRepOperator* and *UsacQualityCheckOperator*. These operators would function effectively over massively parallel processing.

What if the pipeline(s) were run on a daily basis by 7am?

The pipeline has been designed to run by 7am, on a daily basis, as is. As is, the data amount is small enough that the pipeline is completed under 30mins for each each DAG run.

What if the database needed to be accessed by 100+ people?

If the database required 100+ users to simultaneously access the data, I may use high availability between a read/write database and a read-only database. This would give the system freedom to operate, while the updates occur through the pipeline. It also splits requests between two end points, spreading out request traffic.

Appendix

Data Dictionary

***Schema:** usac* ***Table:** erate_470forms*

	Field Name	Data Type	Required	Description	Source
P	application_number	NUMERIC	YES	Unique application number for filing the FCC Form 470.	E-rate API: Basic Information
	nickname	TEXT		Nickname given to FCC Form 470 application.	E-rate API: Basic Information
	funding_year	NUMERIC		Funding year for the FCC Form 470.	E-rate API: Basic Information
	billing_entity_number	TEXT	YES	Unique number given to the billed entity (e.g. school, library, etc)	E-rate API: Basic Information
	fcc_form_status	TEXT		Current status of the FCC Form 470.	E-rate API: Basic Information
	allowable_contract_date	DATE		Earliest date an applicant can sign a contract for contracted services.	E-rate API: Basic Information
	category_one_description	TEXT		Description of Internet Access and/or Telecommunications applicants.	E-rate API: Basic Information
	category_two_description	TEXT	YES	Description of Internal Connections and/or Broadband Services applicants.	E-rate API: Basic Information
	rfp_id	TEXT		Indicates whether a RFP document was provided.	E-rate API: Basic Information
	government_restrictions	TEXT		Indicates if government bidding requirements apply to the sought services.	E-rate API: Basic Information
	government_restriction_descriptions	TEXT		Description of the government bidding requirements applied to the sought services.	E-rate API: Basic Information
	is_statewide	TEXT		Indicates whether the FCC Form 470 is for the entire state.	E-rate API: Basic Information
	is_statewide_public_schools	YES/NO		Indicates whether the FCC Form 470 is for all public schools/districts in the entire state.	E-rate API: Basic Information
	is_statewide_nonpublic_schools	YES/NO		Indicates whether the FCC Form 470 is for all non-public schools/districts in the entire state.	E-rate API: Basic Information
	is_statewide_libraries	YES/NO		Indicates whether the FCC Form 470 is for all libraries in the entire state.	E-rate API: Basic Information
	creation_date	DATETIME		Date/time the FCC Form 470 was created.	E-rate API: Basic Information
	created_by	TEXT		Individual who created the FCC Form 470.	E-rate API: Basic Information
	modified_date	DATETIME	YES	Date/time the FCC Form 470 was modified.	E-rate API: Basic Information
	modified_by	TEXT		Individual who modified the FCC Form 470.	E-rate API: Basic Information
	has_applicable_requests	YES/NO		Indicates whether one or more services requested can are actionable by Broadband.	Calculated Column
	service_request_count	NUMERIC	YES	Provides a count of actionable services requested.	Calculated Column
	estimated_dollar_value	NUMERIC	YES	Cumulative dollar value of actionable services requested.	Calculated Column

Schema: *usac* ***Table:*** *erate_470form_requesters*

Field Name		Data Type	Required	Description	Source
P	application_request_uid	TEXT	YES	Unique ID generated for the service requests.	Calculated Column
F	application_number	NUMERIC	YES	Unique application number for filing the FCC Form 470.	E-rate API: Basic Information
F	billing_entity_number	TEXT	YES	Unique number given to the billed entity (e.g. school, library, etc)	E-rate API: Basic Information
	contact_name	TEXT		Name of the main contact provided on the FCC Form 470.	E-rate API: Basic Information
	contact_phone	TEXT		Phone number of the application’s main contact.	E-rate API: Basic Information
	contact_phone_extension	TEXT		Phone number extension of the application’s main contact.	E-rate API: Basic Information
	contact_email	TEXT		Email address of the main contact.	E-rate API: Basic Information
	technical_name	TEXT		Name of the technical contact provided on the FCC Form 470.	E-rate API: Basic Information
	technical_title	TEXT		Job title of the technical contact provided on FCC Form 470.	E-rate API: Basic Information
	technical_phone	TEXT		Phone number of the technical contact provided on the FCC Form 470.	E-rate API: Basic Information
	technical_phone_extension	TEXT		Phone extension of the technical contact provided on the FCC Form 470.	E-rate API: Basic Information
	technical_email	TEXT		Email address of the technical contact provided on the FCC Form 470.	E-rate API: Basic Information
	authority_name	TEXT		Name of the person authorized to certify the FCC Form 470.	E-rate API: Basic Information
	authority_phone	TEXT		Phone number of the person authorized to certify the FCC Form 470.	E-rate API: Basic Information
	authority_phone_extension	TEXT		Phone extension of the person authorized to certify the FCC Form 470.	E-rate API: Basic Information
	authority_title	TEXT		Title of the person authorized to certify the FCC Form 470.	E-rate API: Basic Information
	authority_email	TEXT		Email address of the person authorized to certify the FCC Form 470.	E-rate API: Basic Information
	authority_employer	TEXT		Employer of the person authorized to certify the FCC Form 470.	E-rate API: Basic Information

Schema: *business* ***Table:*** *sales_representatives*

Field Name		Data Type	Required	Description	Source
P	sales_rep_id	TEXT	YES	9-digit unique ID assigned to the sales representative.	User Input
	first_name	TEXT		Sales representative’s first name.	User Input
	last_name	TEXT		Sales representative’s last name.	User Input
	email_address	TEXT		Sales representative’s email address.	User Input

Schema: *usac* ***Table:*** *billing_entities*

	Field Name	Data Type	Required	Description	Source
P	entity_number	TEXT	YES	Unique number given to the billed entity (e.g. school, library, etc.)	E-rate API: Basic Information
	entity_name	TEXT		Name of the billed entity.	E-rate API: Basic Information
	fcc_registration_number	TEXT		Unique 10-digit number assigned by the FCC to a business or individual that registers with the FCC.	E-rate API: Basic Information
	organization_type	TEXT		The type of the entity including school district, school, library system, library, consortium, and non-instructional facility.	E-rate API: Basic Information
	organization_status	TEXT		Indicates if the organization is active (open) or closed.	E-rate API: Basic Information
	applicant_type	TEXT		Applicant type including: school, school district, library, library system, or consortium.	E-rate API: Basic Information
	eligible_entities	NUMERIC		Number of entities eligible for the services requested.	E-rate API: Basic Information
	website_url	TEXT		Website for the billed entity.	E-rate API: Basic Information
	address_line_1	TEXT		Applicant street address line 1.	E-rate API: Basic Information
	address_line_2	TEXT		Applicant street address line 2.	E-rate API: Basic Information
	city	TEXT		Applicant city.	E-rate API: Basic Information
	state	TEXT		Applicant state.	E-rate API: Basic Information
	zip_code	TEXT		Applicant zip code.	E-rate API: Basic Information
	zip_code_extnsion	TEXT		Applicant zip code extension.	E-rate API: Basic Information
	latitude	NUMERIC		Latitude of the applicant.	E-rate API: Basic Information
	longitude	NUMERIC		Longitude of the applicant.	E-rate API: Basic Information

Schema: *business* ***Table:*** *us_states*

	Field Name	Data Type	Required	Description	Source
P	state_abbreviation	TEXT	YES	Abbreviation for a US state.	User Input
	state_name	TEXT	YES	Full name for a US state.	User Input

Schema: *business* ***Table:*** *sales_state_assignment*

Field Name		Data Type	Required	Description	Source
	team_letter	TEXT	YES	Internal team indicator.	User Input
P,F	state_abbreviation	TEXT	YES	Abbreviation for a US state.	User Input
P,F	sales_rep_id	TEXT	YES	9-digit unique ID assigned to the sales representative.	User Input
	low_job_value	NUMERIC	YES	Lower boundary of sales opportunity values the representative is responsible for.	User Input
	hight_job_value	NUMERIC	YES	Upper boundary of sales opportunity values the representative is responsible for.	User Input

Schema: *usac* ***Table:*** *erate_470form_requests*

Field Name		Data Type	Required	Description	Source
P	service_request_id	TEXT	YES	Unique ID for the service requested.	Calculated Column
F	application_number	NUMERIC	YES	Unique application number for filing the FCC Form 470.	E-rate API: Services Requested
	funding_year	NUMERIC		Funding year for the FCC Form 470.	E-rate API: Services Requested
	service_type	TEXT		Funding request number service type.	E-rate API: Services Requested
	function	TEXT		Indicates the function of the funding request number line item service or product.	E-rate API: Services Requested
	applicable_entities	NUMERIC		Number of entities served by the service requested.	E-rate API: Services Requested
	quantity	NUMERIC		Amount of units of the specific service requested.	E-rate API: Services Requested
	unit	TEXT		Unit type of the service requested (e.g. each, circuits, lines, etc.)	E-rate API: Services Requested
	manufacturer	TEXT		Requested manufacturer or equivalent.	E-rate API: Services Requested
	min_capacity	TEXT		OBSOLETE - Minimum capacity of units desired for the service requested.	E-rate API: Services Requested
	max_capacity	TEXT		OBSOLETE - Maximum capacity of units desired for the service requested.	E-rate API: Services Requested
	needs_installation	YES/NO		Indicates if the service requires installation or initial configuration.	E-rate API: Services Requested
	needs_support	YES/NO		Indicates if the applicant is seeking maintenance or technical support.	E-rate API: Services Requested
	has_applicable_request	YES/NO	YES	Indicates whether this service request is actionable by Broadband.	Calculated Column
	estimated_dollar_value	NUMERIC	YES	Estimated dollar value for the service requested.	Calculated Column

Schema: *business* ***Table:*** *sales_erate_470forms_assignment*

Field Name		Data Type	Required	Description	Source
P	application_number	NUMERIC	YES	Unique application number for filing the FCC Form 470.	Derived Column
F	sales_rep_id	TEXT	YES	9-digit unique ID assigned to the sales representative.	Derived Column
F	billing_entity_number	TEXT	YES	Unique number given to the billed entity (e.g. school, library, etc.)	Derived Column
	state	TEXT	YES	Abbreviation for a US state.	Derived Column
	estimated_dollar_value	NUMERIC	YES	Cumulative dollar value of actionable services requested.	Derived Column