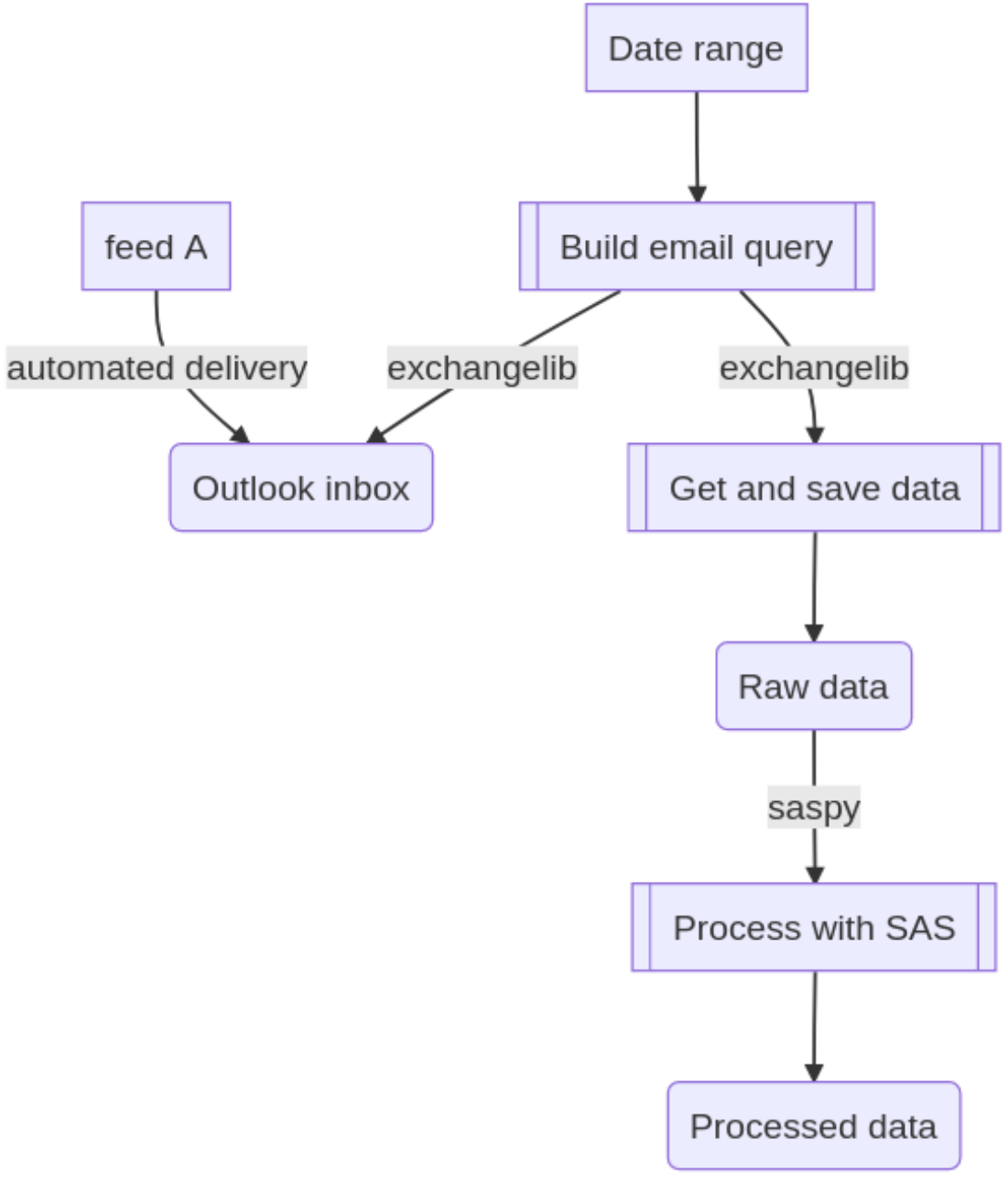
Bind straggling scripts and manual steps into pipelines



RAP your way to success - obtain and process

exchangelib obtain email data

```
# Query to get weekly automated emails from POAL
# start, end are pre-defined date ranges
weekly_poal_query = figs_account.inbox.filter(
    datetime_received__range=(start, end),
    sender="BI-DWSupport@poal.co.nz",
    has_attachments=True)

# Go through all results from email query
for i, email in enumerate(weekly_poal_query):
    # loop through all attachment
    for attachment in email.attachments:

        # if attachment, then save file
        if isinstance(attachment, ex.FileAttachment):

            # Filename and data path
            filename = "weekly_poal_data" + date.today() + ".zip"
            local_path = os.path.join(data_path, "zip_files", filename)

            # Save attachment
            with open(local_path, 'wb') as f:
                f.write(attachment.content)
```

saspy run sas code from python

```
# open list of new data filenames after downloading from Outlook
files_list = open('..\sas_code\combine\files_list.sas').read()

# open data processing code chunks
with open(r'..\sas_code\combine\poal_combine_sas.yaml') as file:
    sas_code = yaml.full_load(file)

# create code blocks by function
preamble = sas_code['preamble']
coarri_combine = sas_code['coarri_combine']
codeco_combine = sas_code['codeco_combine']

# create sas query - concat code blocs
sas_query = f"""{preamble}{files_list}{codeco_combine}"""
res = sas.submit(sas_query)
```

RAP your way to success - document and execute

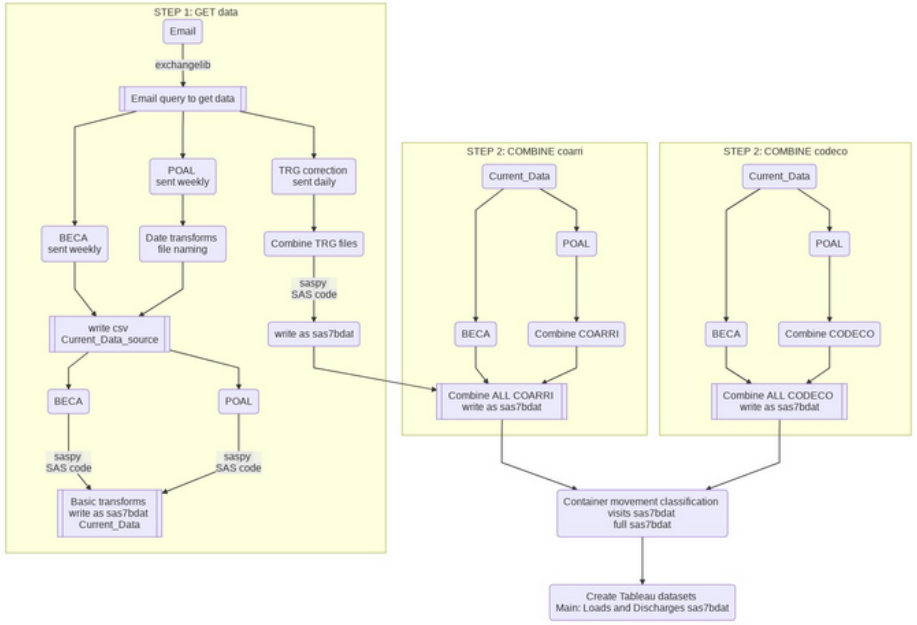
jupyter notebooks create and document pipelines

Introduction

This executable analysis takes the FIGS analyst through the different steps of the update process:

- Step 1: Getting data from email and some basic processing
- Step 2: Combining coarri and codeco
- Final step: movement classification and creating tableau datasets

To modify the diagram below, add the alphanumeric serialisation after https://mermaid-js.github.io/mermaid-live-editor/#/edit/



Setup

This section sets up the connection between Python and SAS processes and allows Python to interact with SAS datasets in addition to pushing code to SAS and pulling results from SAS. The `saspy` library is required. The setup and introduction are covered in the notebook `Introduction to saspy.ipynb` in this repo.

```
# Module imports
import saspy
import pandas as pd
import yaml
import os
import sys
import re
import datetime as dt
import exchangelib as ex
from importlib import reload
```

RAPping in the public sector: binding your legacy code into a pipeline with Python

Ministry of Transport (MOT)
Shrividya Ravi (Shriv)
s.ravi@transport.govt.nz
shriv-portfolio.netlify.app
github.com/shriv