Unlocking the Numbers of International Students in Canada

Data Preparation for Tableau Dashboard "Unlocking the Numbers of International Students in Canada"

Datasource : Temporary Residents: Study Permit Holders – Monthly IRCC Updates - Canada

https://open.canada.ca/data/en/dataset/90115b00-f9b8-49e8-afa3-b4cff8facaee

Here I am going to used 3 dataset (CSV) to create my data analysis & visualization

1. Study permit holders by country of citizenship and year in which permit(s) became effective.[(Link)](<https://www.ircc.canada.ca/opendata-donneesouvertes/data/ODP-TR-Study-IS_CITZ.csv>)
2. Study permit holders by study level, province/territory and year in which permit(s) became effective.[(Link)](<https://www.ircc.canada.ca/opendata-donneesouvertes/data/ODP-TR-Study-IS_PT_study.csv>)
3. Study permit holders by province/territory, gender and year in which permit(s) became effective.[(Link)](<https://www.ircc.canada.ca/opendata-donneesouvertes/data/ODP-TR-Study-IS_PT_gender.csv>)

Let’s walk through building a complete serverless data pipeline on AWS, step-by-step. This pipeline will automate the process of downloading data files, processing them, and saving them in an S3 bucket using AWS Lambda, S3, and EventBridge for monthly scheduling. I’ll guide you through the setup, and we’ll create the Terraform code as we go.

# Pipeline Overview

1. **EventBridge**: Triggers the pipeline every month.
2. **Lambda Function**: Downloads the files, processes the data, and saves it to S3.
3. **S3 Bucket**: Stores the processed files.
4. **Terraform**: Automates resource provisioning.

# Step-by-Step Instructions

## Step 1: Create an S3 Bucket for Processed Files

1. Go to the **S3 console** in AWS.
2. Click **Create bucket** and set a unique bucket name (e.g., my-data-pipeline-bucket).
3. Configure the bucket settings as needed (e.g., keep defaults) and create the bucket.
4. **Note the bucket name** as we’ll use it in our Lambda function

## Step 2: Write Lambda Function Code

The Lambda function will:

1. Download the CSV files from specified URLs.
2. Process the files (e.g., clean and save).
3. Upload them to the S3 bucket.

Save the Lambda function code as lambda\_function.py. Here’s a modified version to handle downloading and uploading to S3:

import requests

import boto3

import os

# URLs of datasets

urls = {

"Study\_citizenship": "https://www.ircc.canada.ca/opendata-donneesouvertes/data/ODP-TR-Study-IS\_CITZ.csv",

"Study\_province\_level": "https://www.ircc.canada.ca/opendata-donneesouvertes/data/ODP-TR-Study-IS\_PT\_study.csv",

"Study\_province\_gender": "https://www.ircc.canada.ca/opendata-donneesouvertes/data/ODP-TR-Study-IS\_PT\_gender.csv"

}

s3\_client = boto3.client('s3')

bucket\_name = os.environ['BUCKET\_NAME']

def download\_file(url):

response = requests.get(url)

response.raise\_for\_status()

return response.content

def lambda\_handler(event, context):

for name, url in urls.items():

data = download\_file(url)

s3\_key = f"{name}.csv"

s3\_client.put\_object(Bucket=bucket\_name, Key=s3\_key, Body=data)

print(f"Uploaded {s3\_key} to S3 bucket {bucket\_name}")

## Step 3: Package the Lambda Code (for Upload)

1. **Install requests library** (Lambda doesn’t include it by default)

pip install requests -t .

Instead of downloading python pandas or numby we can use AWS SDK ARN, those are already created and add as a layer.

pip install pandas -t .

To set up AWS Data Wrangler’s Lambda layer for pandas and numpy, follow these steps:

**Step 1: Get the Latest Lambda Layer ARN for AWS Data Wrangler**

1. Visit the [AWS Data Wrangler GitHub page for Lambda layers](https://github.com/awslabs/aws-data-wrangler) to find the latest Lambda Layer ARNs.
2. In the README or documentation, look for the **Lambda Layer ARN** section.
3. Find the ARN for your **AWS region** and **Python runtime version** (e.g., Python 3.9).

For example, an ARN might look like this for us-east-1:

Example: arn:aws:lambda:us-east-1:336392948345:layer:AWSDataWrangler-Py3-39:latest

**Step 2: Add the Layer to Your Lambda Function**

1. **Go to the Lambda Console** in AWS.
2. Select your Lambda function.
3. Scroll down to the **Layers** section.
4. Click on **Add a layer**.
5. Choose **Specify an ARN**.
6. Paste the ARN you found from the AWS Data Wrangler GitHub page.
7. Click **Add** to attach the layer to your function.

### Step 3: Test the Lambda Function

After adding the layer, your Lambda function should be able to import pandas and numpy without any issues. You can add a simple test function to ensure everything is set up correctly:

python

Copy code

import pandas as pd

import numpy as np

def lambda\_handler(event, context):

# Simple test to ensure pandas and numpy work

df = pd.DataFrame(np.random.randn(5, 3), columns=['A', 'B', 'C'])

print(df)

return {

'statusCode': 200,

'body': "Pandas and NumPy are working!"

}

### Step 4: Adjust Code as Needed

Once confirmed, you can proceed with your intended pandas processing code in your Lambda function, knowing that the pandas and numpy dependencies are supported by the Lambda layer.

### Important Note

The latest version tag in the ARN is updated frequently. It’s generally best to specify the exact version instead of using :latest to ensure version consistency across deployments.

1. **Zip the files**:

zip -r lambda\_function.zip .