AWS – LAMDA

import boto3

import csv

import io

import os

import json

import pymysql

def lambda\_handler(event, context):

dynamodb = boto3.resource('dynamodb')

table = dynamodb.Table('FileProcessingStatus') # DynamoDB table name

required\_files = ['WBSwithIO.csv', 'CostCenterData.csv', 'ProjectLists.csv']

all\_files\_processed = True

# Database connection

rds\_host = os.environ['DB\_HOST']

rds\_user = os.environ['DB\_USER']

rds\_password = os.environ['DB\_PASSWORD']

rds\_db\_name = os.environ['DB\_NAME']

conn = pymysql.connect(host=rds\_host, user=rds\_user, passwd=rds\_password, db=rds\_db\_name)

try:

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

for record in event['Records']:

bucket\_name = record['s3']['bucket']['name']

file\_name = record['s3']['object']['key']

# Read CSV file from S3

response = s3\_client.get\_object(Bucket=bucket\_name, Key=file\_name)

file\_content = response['Body'].read().decode('utf-8')

csv\_reader = csv.reader(io.StringIO(file\_content))

# Determine the table name based on the file name

table\_name = ""

if 'WBSwithIO' in file\_name:

table\_name = 'wbs\_with\_io'

elif 'CostCenterData' in file\_name:

table\_name = 'cost\_center\_data'

elif 'ProjectLists' in file\_name:

table\_name = 'project\_lists'

# Skip header and insert data into the database

next(csv\_reader)

with conn.cursor() as cursor:

for row in csv\_reader:

query = f"INSERT INTO {table\_name} VALUES ({', '.join(['%s' for \_ in row])})"

cursor.execute(query, row)

conn.commit()

# Update the DynamoDB table

table.update\_item(

Key={'FileName': file\_name},

UpdateExpression='SET #P = :val',

ExpressionAttributeNames={'#P': 'Processed'},

ExpressionAttributeValues={':val': True}

)

# Check if all files are processed

response = table.scan()

for item in response['Items']:

if item['FileName'] in required\_files and not item.get('Processed', False):

all\_files\_processed = False

break

# If all files are processed, perform data cleaning within this Lambda

if all\_files\_processed:

with conn.cursor() as cursor:

cursor.execute("SET SQL\_SAFE\_UPDATES = 0;")

cursor.execute("ALTER TABLE cost\_center\_data ADD COLUMN object\_doc\_no VARCHAR(255);")

cursor.execute("UPDATE cost\_center\_data SET object\_doc\_no = CONCAT(Object, Document\_Number);")

cursor.execute("ALTER TABLE wbs\_with\_io ADD COLUMN partner\_object\_doc\_no VARCHAR(255);")

cursor.execute("UPDATE wbs\_with\_io SET partner\_object\_doc\_no = CONCAT(Partner\_Object, Document\_Number);")

cursor.execute("ALTER TABLE cost\_center\_data ADD COLUMN WBS\_Element VARCHAR(60) AFTER Period;")

cursor.execute("UPDATE cost\_center\_data SET WBS\_Element = Partner\_object;")

cursor.execute("CREATE TABLE cc\_deletes AS SELECT \* FROM cost\_center\_data WHERE 1=0;")

cursor.execute("""

INSERT INTO cc\_deletes

SELECT ccd.\*

FROM cost\_center\_data ccd

LEFT JOIN wbs\_with\_io wio ON ccd.object\_doc\_no = wio.partner\_object\_doc\_no

WHERE wio.Object IS NULL;

""")

cursor.execute("""

CREATE TABLE wbs\_cc AS

SELECT Full\_Name, Personnel\_Number, Total\_Quantity, Val\_COArea\_Crcy, Period, WBS\_Element, Object, Partner\_Object, CO\_Object\_Name, Cost\_Element\_Descr, Document\_Number, ParActivity, Fiscal\_Year, Cost\_Element, Functional\_Area, Posting\_Date

FROM wbs\_with\_io

UNION ALL

SELECT Full\_Name, Personnel\_Number, Total\_Quantity, Val\_COArea\_Crcy, Period, WBS\_Element, Partner\_Object AS Object, Partner\_Object, CO\_Object\_Name, Cost\_Element\_Name AS Cost\_Element\_Descr, Document\_Number, Activity\_Type AS ParActivity, Fiscal\_Year, Cost\_Element, Functional\_Area, Posting\_Date

FROM cc\_deletes;

""")

cursor.execute("""

CREATE TABLE final\_clean\_data AS

SELECT \*

FROM wbs\_cc

WHERE Cost\_Element\_descr NOT IN ('Settle IO-MS(NP)', 'Settle IO-REM.COS (NP)', 'Settle IO-RD(NP)', 'Settle IO-REM COS')

AND Object NOT LIKE '9%'

AND Partner\_Object NOT LIKE 'PA15%'

AND Cost\_Element\_descr <> 'Internal Labor';

""")

conn.commit()

# Reset the DynamoDB table for the next run

for file\_name in required\_files:

table.update\_item(

Key={'FileName': file\_name},

UpdateExpression='SET #P = :val',

ExpressionAttributeNames={'#P': 'Processed'},

ExpressionAttributeValues={':val': False}

)

finally:

# Ensure the database connection is closed

conn.close()

return {

'statusCode': 200,

'body': 'File processing and data cleaning completed.'

}

This is the aws lambda function, when a file is uploaded to the s3 bucket this lambda function is triggered. Each file upload is triggering the function so concurrently it starts uploading the files to the table from 3 different executions. However, before the start of the cleaning dynamodb is looking if all the uploads are finished, if so, it starts cleaning. If not it still waits for the others. After each file this is checked. Since three of them simultaneously start, the first two ones that are finishing their uploads will see it is not ready but the last one will see after it is ready, all of them are ready so it will also execute the data cleaning. Hence in every three-file upload which is our system based on, we will have a cleaning. This lambda function takes care of everything in terms of ETL.