Case Study: Data Ingestion and Data Manipulation

# Scenario Overview

Every day, four files are received via email as an attachment for previous day data.  
1. Three files represent daily order details (including signups, transfers, and churn events).  
2. One file contains a list of active services as of the previous day.  
  
You are required to automate the extraction, processing, and ingestion of this data into raw tables, followed by building a flattened table in SQL for further analysis. User should be able to see the service events and status in one view.

## Part 1: Python Task

You are required to develop a Python script that performs the following:

1. Fetch Emails from Outlook:  
 - The script should connect to your Outlook inbox and download emails from a specific folder.  
 - Extract the attached file from the email.

2. Process the Excel File:  
 - Read the 4 files using Python’s pandas’ library.  
 - Orders: Combine and clean the data from the 3 files (Service, Customer, Order). Take only those that can be mapped with Service file.  
 - Actives: Process the data from the active file to capture the list of active services as of the previous day.  
 - Perform basic data cleaning, including handling missing values, adjusting column data types, and any necessary formatting to prepare the data for ingestion.

3. Ingest Data into Raw Tables:  
 - Write each file into a corresponding raw table within your database. You should create the following tables:

1. Order Table

2. Active Service Table  
 - SQL Alchemy, NZ Alchemy are not supported by Netezza. Please think through how you will be able to ingest the data to the database.

## Part 2: SQL Task

Using the raw tables created in the previous step, write SQL queries to perform the following:

1. Build a Flattened Table for Analysis:  
 - Join and aggregate the data from the order table and the active service table to create a flattened table that will be used for analysis.

- Flattened table will be a snapshot of orders and status of the service on a daily, weekly and monthly frequency  
 - This table should be able to track:  
 - Whether a service is a new signup.  
 - Whether a service has made a churn event.

- Whether the new signup is a transfer  
 - The current status of the service (e.g., active or inactive)

# Deliverables

1. Python Code:  
 - A script that automatically fetches the daily file, processes the worksheets, and ingests the data into the database’s raw tables.

2. SQL Scripts:  
 - SQL queries that create the necessary flattened table for analysis.