2. MANIPULACIÓN DE DATOS

Importar las bibliotecas necesarias:

i*mport pandas as pd*

*import numpy as np*

Se importan las librerías pandas para manejar datos tabulares, numpy para operaciones matemáticas

Leer el archivo de Excel y mostrar las primeras filas:

df = pd.read\_excel('/content/Datos Maestros VF.xlsx')

df.head()

Se lee un archivo Excel llamado 'Datos Maestros VF.xlsx' y se almacenan los datos en un DataFrame llamado "df". Luego, se utiliza el método "head()" para mostrar las primeras filas del DataFrame, lo que nos permite tener una vista previa de los datos.

Definir las columnas a eliminar:

columnas\_elim = ['Precio de Arranque (Par)', 'Mínimo técnico por central', 'Inflexibilidad por unidad para generar AGC (QA-)', 'Minimo Operativo por central (MO del Ofei)', 'CENTRAL (OFEI)', 'UNIDAD (OFEI)', 'Nombre visible Central']

Se crea una lista llamada "columnas\_elim" que contiene los nombres de las columnas que se desean eliminar del DataFrame "df".

Eliminar las columnas no deseadas y mostrar las primeras filas del DataFrame resultante:

df\_pros = df.drop(columns=columnas\_elim)

df\_pros = df\_pros.drop('Minimo ténico por unidad', axis=1)

df\_pros.head()

Se crea un nuevo DataFrame llamado "df\_pros" eliminando las columnas especificadas en "columnas\_elim" utilizando el método "drop()". Luego, se elimina otra columna adicional llamada "Minimo ténico por unidad" utilizando nuevamente el método "drop()". Por último, se muestra una vista previa del DataFrame resultante utilizando el método "head()".

Leer el archivo CSV, realizar operaciones y limpiar datos:

ruta\_archivo = '/content/dDEC1204.TXT'

df\_dec\_central = pd.read\_csv(ruta\_archivo, encoding='latin-1', delimiter=',', quotechar='"', header=None)

df\_dec\_central['Total'] = df\_dec\_central.iloc[:, 2:].sum(axis=1)

df\_dec\_central = df\_dec\_central[df\_dec\_central['Total'] > 0]

df\_dec\_central = df\_dec\_central.drop(columns=df\_dec\_central.columns[1:25])

Se lee un archivo CSV llamado 'dDEC1204.TXT' y se almacenan los datos en un DataFrame llamado "df\_dec\_central". Se realiza una operación para calcular la suma de las columnas desde la columna 2 hasta el final (fila 1) y se agrega el resultado en una nueva columna llamada 'Total'. Luego, se eliminan las filas que tienen un valor 'Total' igual a cero. Por último, se eliminan las columnas desde la columna 1 hasta la columna 24 utilizando el método "drop()".

Renombrar las columnas del DataFrame "df\_dec\_central":

column\_nom = {0: 'CENTRAL (dDEC, dSEGDES, dPRU…)', 'Total': 'Total de horas'}

df\_dec\_central = df\_dec\_central.rename(columns=column\_nom)

Se crea un diccionario llamado "column\_nom" que asigna nuevos nombres a las columnas del DataFrame "df\_dec\_central". Luego, se utiliza el método "rename()" para cambiar los nombres de las columnas según el diccionario.

Mostrar las últimas filas del DataFrame "df\_dec\_central":

df\_dec\_central.head()

print('=' \* 64)

df\_dec\_central.tail(5)

Se muestra una vista previa de las primeras filas del DataFrame "df\_dec\_central" con el método "head()". Luego, se imprime una línea de separación de "=" repetida 64 veces para delimitar la salida. Finalmente, se muestran las últimas 5 filas del DataFrame "df\_dec\_central" con el método "tail()".

Realizar la fusión (merge) de los DataFrames "df\_pros" y "df\_dec\_central":

df\_merge = pd.merge(df\_pros, df\_dec\_central, on='CENTRAL (dDEC, dSEGDES, dPRU…)')

df\_merge.head()

Se realiza una fusión (merge) de los DataFrames "df\_pros" y "df\_dec\_central" utilizando el método "merge()" y se especifica que la columna utilizada para la fusión es 'CENTRAL (dDEC, dSEGDES, dPRU…)'. Luego, se muestra una vista previa del DataFrame resultante "df\_merge" con el método "head()".

Verificar si hay valores de 0 en la columna 'Total de horas':

if (df\_merge['Total de horas'] == 0).any():

print("El DataFrame contiene al menos un valor 0")

else:

print("El DataFrame no contiene valores 0")

Se verifica si hay algún valor igual a cero en la columna 'Total de horas' del DataFrame "df\_merge" utilizando una condición booleana y el método "any()". Si se encuentra al menos un valor cero, se imprime un mensaje indicando que el DataFrame contiene al menos un valor cero. De lo contrario, se imprime un mensaje indicando que no hay valores cero en el DataFrame.

Crear un nuevo DataFrame "df" y guardar en archivos CSV y Excel:

df = pd.DataFrame(df\_merge)

df.to\_csv('Prueba2.csv', index=False)

df.to\_excel('Prueba-2.xlsx', index=False)

Se crea un nuevo DataFrame llamado "df" a partir del DataFrame "df\_merge". Luego, se guarda este DataFrame en archivos CSV y Excel con los nombres 'Prueba2.csv' y 'Prueba-2.xlsx', respectivamente, utilizando los métodos "to\_csv()" y "to\_excel()".

Verificar el tipo de objeto del DataFrame "df":

type(df)

Se verifica el tipo de objeto del DataFrame "df" utilizando la función "type()". La salida mostrará el tipo de objeto, que en este caso es un DataFrame de pandas.