

**Nombre:** Josselyn Fabiola Duarte Torres

## Carga y cruce de información de personas

Desarrollar un flujograma, donde se recibe un archivo Excel que contiene: nombre completo de persona (de la forma "Juan Antonio Garcia Hernandez"), número de DUI y otros campos demográficos.

Los nombres deben ser clasificados como: primer nombre, segundo nombre, tercer nombre, primer apellido, segundo apellido, apellido de casada.

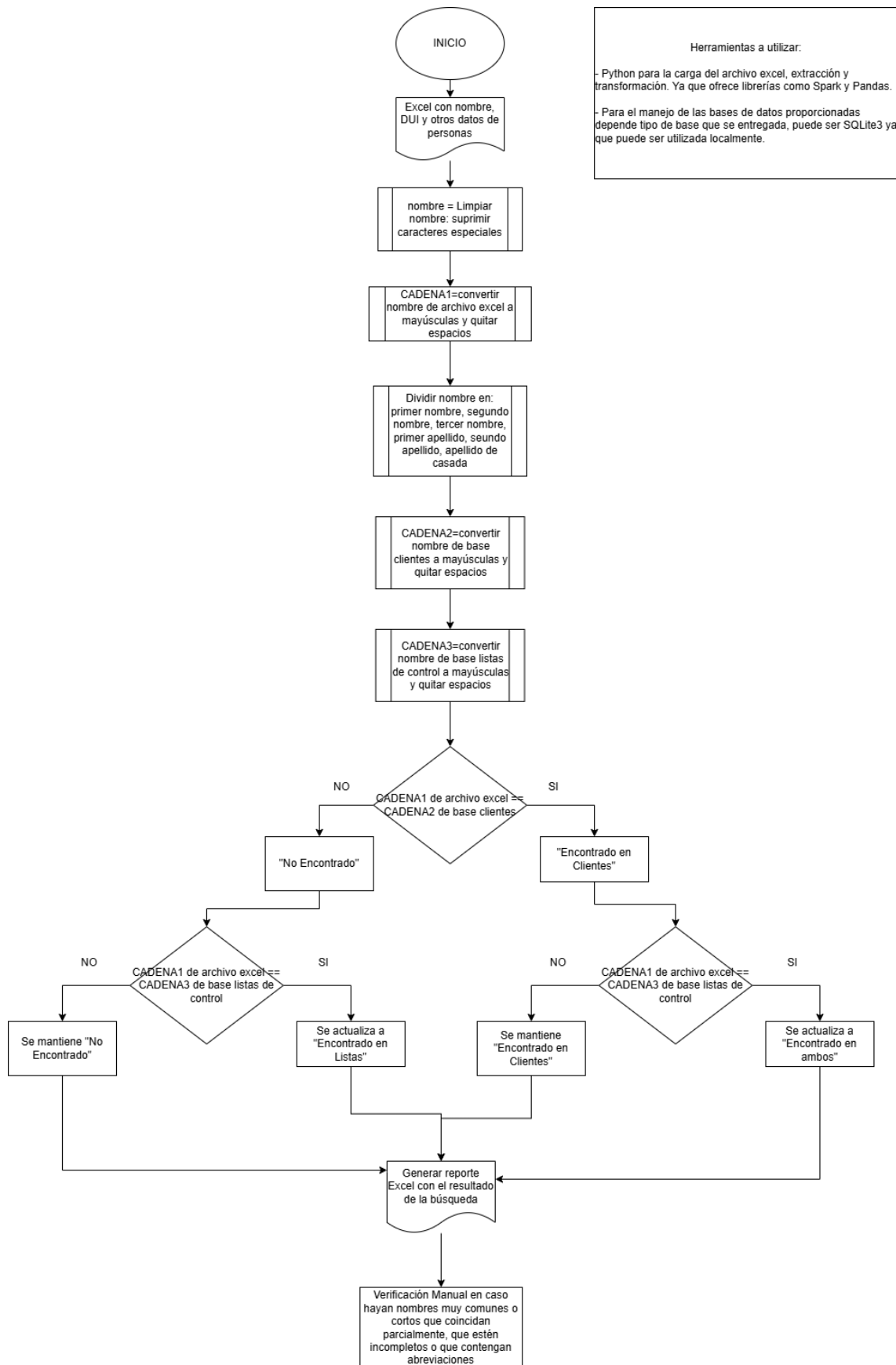
Se requiere un reporte Excel donde estén clasificados los registros del archivo original indicando si el registro fue encontrado en alguna base interna, ya sea como cliente de banco, como persona en lista de control, ambos casos, o indicar si no encontró en ninguna (registro nuevo). Realice operaciones de limpieza de cadenas para maximizar la efectividad del cruce masivo.

Cartera de clientes del banco: información general de los clientes.

Lista de control: información de personas con las que el banco no debería establecer relación.

Consideraciones:

- \* Plantear que herramientas se utilizarían.
- \* El nombre completo puede tener caracteres especiales, espacios en blanco, números, mayúsculas, minúsculas.
- \* Usar expresiones regulares de ser necesario.
- \* Indicar si se detecta alguna inconsistencia. Es decir, comente si existen escenarios ambiguos en el resultado del proceso que deban ser verificados manualmente.



## Realizar el siguiente ETL

Realizar un ETL que cargue en una tabla el archivo "fuente.xlsx"

Se requiere un reporte Excel donde estén clasificados las personas basadas en la cartera donde encontró (tabla clientes, tabla lista\_control) o indicar si no encontró en ninguna cartera.

Consideraciones:

- \* Utilizar las tablas de la base sqlite proporcionada (ev\_ing\_dat.db)
- \* Usar python, spark, sql, Power BI, según considere necesario.
- \* Documentar la solución, los pasos del proceso.
- \* Usa bitácora para el proceso (log y tabla de bitácora)
- \* Versionar la solución usando git.
- \* Calendarizar el programa para que se ejecute todos los 1° de cada mes a las 10 a.m.

**Reporte esperado:**

Nombre completo	Primer nombre	Segundo nombre	Tercer nombre	Primer apellido	Segundo apellido	Apellido de casada	Clasificación
Juan Antonio Garcia Hernandez	Juan	Antonio		Garcia	Hernandez		Lista de control
María Lopez de Hernandez	María			López		De Hernandez	Cartera de Clientes
Sofía Gomez	Sofía			Gomez			No se encontró en las BD

Pasos:

1. Primero se creó el núcleo del ETL:

- **Extracción**

Ya que se solicita que el archivo 'fuente.xlsx' se almacene en una tabla, y por ello se cargara en la base 'ev\_ing\_dat.db'. Se hizo el proceso de leer el archivo conectar a la base (ya que ahí se creara la nueva tabla y ahí se encuentran las tablas de clientes y listas) y crear la tabla en caso no exista:

```

#Excel
arcExcel='fuente.xlsx'
fuenteEx=pnds.read_excel(arcExcel)

#Conectar a DB
dataBase='ev_ing_dat.db'
conn=sqlite3.connect(dataBase)
cursor=conn.cursor()

try:
    #Crear tabla si no existe
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS fuente (
            idFuente INTEGER PRIMARY KEY AUTOINCREMENT,
            nombre VARCHAR(250),
            documento VARCHAR(20)
        );
    ''')

```

Cuando ya se tiene la tabla creada se procede a la carga.

```

#Limpiar tabla fuente
fuenteDltQry="DELETE FROM fuente"
cursor.execute(fuenteDltQry)
conn.commit()

seqDltQry="DELETE FROM sqlite_sequence WHERE name = 'fuente'"
cursor.execute(seqDltQry)
conn.commit()

#Cargar excel en tabla
for index, row in fuenteEx.iterrows():
    cursor.execute('''
        INSERT INTO fuente (nombre,documento)
        VALUES(?,?);
        ''',(row['Nombre'],row['documento']))
    conn.commit()

```

## - Transformación

Primero se hizo una limpieza de las cadenas de texto que se iban a comparar, haciendo uso de SQL y funciones de Python.

```
#Clientes
clientesQry="SELECT 'cliente' tabla,UPPER(REPLACE(TRIM(nombre1||nombre2||apellido1||apellido2||apellido_casada),' ','')) nombreCC,nomb
clientes=pnds.read_sql(clientesQry,conn)
clientes['nombreCC'] = clientes['nombreCC'].str.replace(r'^a-zA-Z0-9\s',' ', regex=True)

#Listas
listasQry="SELECT 'lista' tabla,UPPER(REPLACE(TRIM(nombre),' ','')) nombreCC,nombre,UPPER(REPLACE(TRIM(documento),' ','')) documento FR
listas=pnds.read_sql(listasQry,conn)
listas['nombreCC'] = listas['nombreCC'].str.replace(r'^a-zA-Z0-9\s',' ', regex=True)

#Fuente
fuenteQry="SELECT UPPER(REPLACE(TRIM(nombre),' ','')) nombreCC,nombre,' Tercer_Nombre,UPPER(REPLACE(TRIM(documento),' ','')) documento
fuente=pnds.read_sql(fuenteQry,conn)
fuente['nombreCC'] = fuente['nombreCC'].str.replace(r'^a-zA-Z0-9\s',' ', regex=True)
```

Luego se definió la función encargada de comparar las tablas y determinar si la persona existe en una de las bases: clientes o listas o si no fue encontrado.

```
#Busqueda de tabla fuente en tablas clientes y listas
def clasificar(row):
    if row['nombreCC'] in clientes['nombreCC'].values:
        return 'Cartera de Clientes'
    elif row['nombreCC'] in listas['nombreCC'].values:
        return 'Lista de Control'
    else:
        return 'No se encontro en las BD'

try:
    fuente['Clasificacion'] = fuente.apply(clasificar, axis=1)
```

## - Carga

Ya que se debe generar un reporte Excel con el resultado, se usando la librería 'pandas' de Python de la siguiente forma:

```
#Formato y extraccion de Excel
fuente[['Primer_Nombre','Segundo_Nombre','Primer_Apellido','Segundo_Apellido','Apellido_de_Casada']]=fuente['nombre'].str.split(' ',ex
columnasExcel=['nombre','Primer_Nombre','Segundo_Nombre','Tercer_Nombre','Primer_Apellido','Segundo_Apellido','Apellido_de_Casada','Cl
fuenteExcel=fuente[columnasExcel].sort_values(by='Clasificacion',ascending=[True])
fuenteExcel.to_excel('clasificados.xlsx',sheet_name='datos',index=False)
registros=fuenteExcel.shape[0]
estado='Ok'
logsClas('Carga','Se completo la carga de archivo excel')
except Exception as e:
    logsClas('Error',f'Error en carga de archivo: {str(e)}')
    estado=str(e)
```

## 2. Creación de bitácora y logs

Para el proceso de registrar en bitácora y generar logs se hizo lo siguiente:

## - Logs

Se definió una función usando la librería logging que ofrece Python.

```
#Configurando logging
logging.basicConfig(filename='logs/clasificados.log', level=logging.INFO, format='%(asctime)s - %(message)s')
def logsClas(status,message):
    logging.info(f"{status}:{message}")
```

Y a lo largo del código se fue haciendo uso de esta función para registrar en el log los pasos y/o errores.

```
#log
logsClas('Inicio','Inicio del ETL para clasificar personas')
```

```
except Exception as e:
    logsClas('Error',f'Error en creacion y carga de tabla: {str(e)}')
    estado=str(e)
```

Mostrando un resultado como el siguiente.

espaldo > Documentos > Prueba\_Tecnica\_Fabi > evaluación\_externa > logs

Nombre	Fecha de modificación	Tipo
clasificados	8/12/2024 00:23	Docum

Nuevo Documento de texto.txt Nuevo Documento de texto.txt .env clasificados.log

Archivo Editar Ver

```
2024-12-08 00:23:00,010 - Inicio:Inicio del ETL para clasificar personas
2024-12-08 00:23:00,355 - Carga:Se completo la carga de archivo excel
2024-12-08 00:23:00,357 - Fin:Se completo el proceso
2024-12-08 00:23:00,358 - Job "ejecucion (trigger: cron[day='8', hour='0', minute='23'], next run at: 2025-01-08 0
```

## - Bitácora

Para registro en bitácora se creo una tabla y se coloco el insert necesario para indicar el resultado de la ejecución.

```

finally:
    #Crear tabla bitacora si no existe
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS bitacora (
            idFuente INTEGER PRIMARY KEY AUTOINCREMENT,
            nombre VARCHAR(250),
            registros VARCHAR(20),
            estado VARCHAR(200),
            archivo VARCHAR(100),
            fecha VARCHAR(100)
        );
    ''')
    conn.commit()

    #Registro en bitacora
    cursor.execute('''
        INSERT INTO bitacora (nombre,registros,estado,archivo,fecha)
        VALUES(?,?,?,?,?);
    ''',(nombreProceso,registros,estado,archivo,fecha))
    conn.commit()

```

### 3. Calendarización

Para la calendarización se hizo uso de la librería APScheduler de Python.

Se creo una función en la que se colocó todo el código a ejecutarse cada 1er día de cada mes a las 10 am.

```

#Programando ejecución
def ejecucion():
    print("Inicio el cron")

    nombreProceso='Clasificados'
    archivo=os.path.basename(__file__)
    fecha=datetime.now().strftime('%Y-%m-%d %H:%M:%S')
    registros=0

    #Configurando logging
    logging.basicConfig(filename='logs/clasificados.log', level=logging.INFO, format='%(asctime)s - %(message)s')
    def logsClas(status,message):
        logging.info(f"{status}:{message}")

    #log

```

Y se creo el objeto encargado de programar la ejecución.

```

scheduler = BlockingScheduler()
scheduler.add_job(ejecucion, 'cron', day=8, hour=0, minute=23)
scheduler.start()

```

#### 4. Versionamiento usando GIT

Se preparo el repositorio local para hacer uso de GIT en el versionamiento del proyecto.

```
C:\Users\Diego\Documents\Prueba_Tecnica_Fabi\evaluación_externa>git init
Initialized empty Git repository in C:/Users/Diego/Documents/Prueba_Tecnica_Fabi/evaluación_externa/.git/

C:\Users\Diego\Documents\Prueba_Tecnica_Fabi\evaluación_externa>git config --global user.name "Fabiola Duarte"

C:\Users\Diego\Documents\Prueba_Tecnica_Fabi\evaluación_externa>git config --global user.email "josselynfabiola@gmail.com"
```

Se creo un repositorio remoto en GitHub

<https://github.com/fabioladt/PruebaTecnica.git>

Se asocio el repositorio local con el remoto

Y se hicieron los commits en el repositorio remoto

```
C:\Users\Diego\Documents\Prueba_Tecnica_Fabi\evaluación_externa>git commit -m "Iniciando versionamiento en repositorio remoto"
[master (root-commit) 21a88f0] Iniciando versionamiento en repositorio remoto
7 files changed, 273 insertions(+)
create mode 100644 ETL.py
create mode 100644 ETLSpark.py
create mode 100644 clasificados.xlsx
create mode 100644 ev_ing_dat.db
create mode 100644 "evaluaci\303\263n_ing_datos_externo.docx"
create mode 100644 fuente.xlsx
create mode 100644 logs\clasificados.log

C:\Users\Diego\Documents\Prueba_Tecnica_Fabi\evaluación_externa>git push -u origin master
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 12 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (10/10), 81.65 KiB | 16.33 MiB/s, done.
Total 10 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/fabioladt/PruebaTecnica/pull/new/master
remote:
To https://github.com/fabioladt/PruebaTecnica.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
```

fabioladt / PruebaTecnica

<> Code Issues Pull requests Actions Projects Security Insights Settings

PruebaTecnica (Private) Unwatch 1

master had recent pushes 1 minute ago Compare & pull request

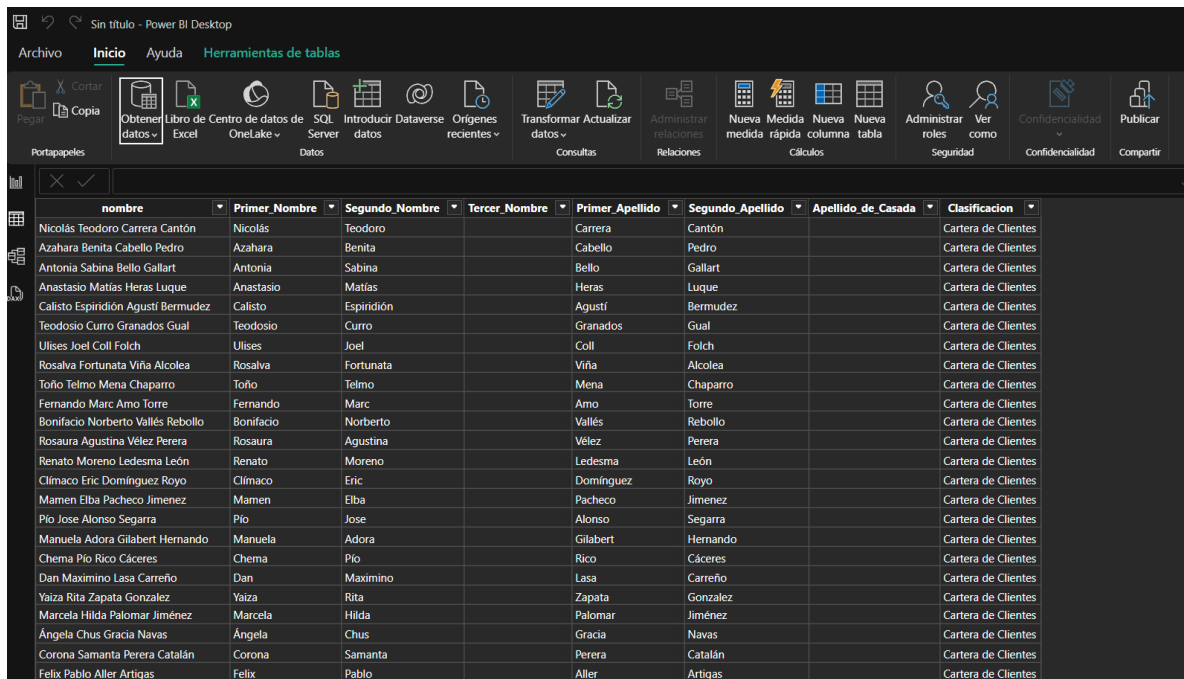
master 2 Branches 0 Tags Go to file Add file Code

This branch is 1 commit ahead of, 1 commit behind main.

fabioladt	Iniciando versionamiento en repositorio remoto	21a88f0 · 1 minute ago	1 Commit
logs	Iniciando versionamiento en repositorio remoto	1 minute ago	
ETL.py	Iniciando versionamiento en repositorio remoto	1 minute ago	
ETLSpark.py	Iniciando versionamiento en repositorio remoto	1 minute ago	



## 5. Resultado



The screenshot shows the Microsoft Power BI Desktop application. The ribbon at the top includes 'Inicio', 'Ayuda', and 'Herramientas de tablas'. The 'Inicio' ribbon has sections for 'Portapapeles' (Pegar, Cortar, Copia), 'Datos' (Obtener datos, Libro de Excel, Centro de datos de OneLake, SQL Server, Introducir Dataverse, Orígenes recientes), 'Consultas' (Transformar datos, Actualizar), 'Relaciones' (Administrar relaciones), 'Cálculos' (Nueva medida, Nueva medida rápida, Nueva columna, Nueva tabla), 'Seguridad' (Administrar roles, Ver como), 'Confidencialidad' (Confidencialidad), and 'Publicar'. Below the ribbon, a data table is displayed with the following columns: nombre, Primer\_Nombre, Segundo\_Nombre, Tercer\_Nombre, Primer\_Apellido, Segundo\_Apellido, Apellido\_de\_Casada, and Clasificación. The table contains 30 rows of data, all of which are categorized as 'Cartera de Clientes' in the 'Clasificación' column.

nombre	Primer_Nombre	Segundo_Nombre	Tercer_Nombre	Primer_Apellido	Segundo_Apellido	Apellido_de_Casada	Clasificación
Nicolás Teodoro Carrera Cantón	Nicolás	Teodoro		Carrera	Cantón		Cartera de Clientes
Azahara Benita Cabello Pedro	Azahara	Benita		Cabello	Pedro		Cartera de Clientes
Antonia Sabina Bello Gallart	Antonia	Sabina		Bello	Gallart		Cartera de Clientes
Anastasio Matías Heras Luque	Anastasio	Matías		Heras	Luque		Cartera de Clientes
Calisto Espiridión Agustí Bermudez	Calisto	Espiridión		Agustí	Bermudez		Cartera de Clientes
Teodosio Curro Granados Gual	Teodosio	Curro		Granados	Gual		Cartera de Clientes
Ulises Joel Coll Folch	Ulises	Joel		Coll	Folch		Cartera de Clientes
Rosalva Fortunata Viña Alcolea	Rosalva	Fortunata		Viña	Alcolea		Cartera de Clientes
Toño Telmo Mena Chaparro	Toño	Telmo		Mena	Chaparro		Cartera de Clientes
Fernando Marc Amo Torre	Fernando	Marc		Amo	Torre		Cartera de Clientes
Bonifacio Norberto Vallés Rebollo	Bonifacio	Norberto		Vallés	Rebollo		Cartera de Clientes
Rosaura Agustina Vélez Perera	Rosaura	Agustina		Vélez	Perera		Cartera de Clientes
Renato Moreno Ledesma León	Renato	Moreno		Ledesma	León		Cartera de Clientes
Climaco Eric Domínguez Royo	Climaco	Eric		Domínguez	Royo		Cartera de Clientes
Mamen Elba Pacheco Jimenez	Mamen	Elba		Pacheco	Jimenez		Cartera de Clientes
Pio Jose Alonso Segarra	Pio	Jose		Alonso	Segarra		Cartera de Clientes
Manuela Adora Gilabert Hernando	Manuela	Adora		Gilabert	Hernando		Cartera de Clientes
Chema Pio Rico Cáceres	Chema	Pio		Rico	Cáceres		Cartera de Clientes
Dan Maximino Lasa Carreño	Dan	Maximino		Lasa	Carreño		Cartera de Clientes
Yaiza Rita Zapata Gonzalez	Yaiza	Rita		Zapata	Gonzalez		Cartera de Clientes
Marcela Hilda Palomar Jiménez	Marcela	Hilda		Palomar	Jiménez		Cartera de Clientes
Ángela Chus Gracia Navas	Ángela	Chus		Gracia	Navas		Cartera de Clientes
Corona Samanta Perera Catalán	Corona	Samanta		Perera	Catalán		Cartera de Clientes
Felix Pablo Aller Artigas	Felix	Pablo		Aller	Artigas		Cartera de Clientes

### Carga de listas de control ONU y OFAC.

Se tienen dos URLs y se requiere un proceso que cargue esta información en diferentes tablas relacionales según considere.

ONU: <https://scsanctions.un.org/resources/xml/en/consolidated.xml>

OFAC: <https://www.treasury.gov/ofac/downloads/sdn.xml>

- Analizar la estructura de los archivos xml y considerar que estos son actualizados de forma periódica.
- Crear las tablas y relaciones que considere necesario.
- Utilizar Python y sqlite.
- Compartir evidencias del funcionamiento, scripts usados (Python, sql, etc)
- Versionar el proceso usando git

1. Se ha creado una base en SQLite para este proceso, en dicha base se crearan las tablas necesarias.

```
def create_db():
    conn=sqlite3.connect('listasOnuOfac.db')
    cursor=conn.cursor()

    cursor.execute('''
        CREATE TABLE IF NOT EXISTS "PersonasONU" (
            "DataID"      TEXT,
            "Version"     TEXT,
            "PrimerNombre" TEXT,
            "SegundoNombre" TEXT,
            "TipoLista"   TEXT,
            "Referencia"   TEXT,
            "FechaLista"   TEXT,
            "Comentarios"  TEXT,
            "Titulo"       TEXT,
            "Designacion"  TEXT,
            "Nacionalidad" TEXT,
            "TipoLista2"   TEXT,
```

2. Se ha definido una función para cargar primero el archivo de la ONU. Primero se hace una limpieza de las tablas

```
response=requests.get(url)
xmlEstructura=xmlET.fromstring(response.content)

conn=sqlite3.connect('listasOnuOfac.db')
cursor=conn.cursor()

cursor.execute("DELETE FROM PersonasONU")
cursor.execute("DELETE FROM PersonasONUAct")
cursor.execute("DELETE FROM PersonasONUAlias")
conn.commit()
```

Luego se hace un recorrido por el XML para extraer la información de cada persona

```
for INDIVIDUAL in xmlEstructura.findall('..//INDIVIDUAL'):
    DataId=INDIVIDUAL.find('DATAID').text if INDIVIDUAL.find('DATAID') is not None else None
    Version=INDIVIDUAL.find('VERSIONNUM').text if INDIVIDUAL.find('VERSIONNUM') is not None else None
    PrimerNombre=INDIVIDUAL.find('FIRST_NAME').text if INDIVIDUAL.find('FIRST_NAME') is not None else None
    SegundoNombre=INDIVIDUAL.find('SECOND_NAME').text if INDIVIDUAL.find('SECOND_NAME') is not None else None
    TipoLista=INDIVIDUAL.find('UN_LIST_TYPE').text if INDIVIDUAL.find('UN_LIST_TYPE') is not None else None
    Referencia=INDIVIDUAL.find('REFERENCE_NUMBER').text if INDIVIDUAL.find('REFERENCE_NUMBER') is not None else None
    FechaLista=INDIVIDUAL.find('LISTED_ON').text if INDIVIDUAL.find('LISTED_ON') is not None else None
    Comentarios=INDIVIDUAL.find('COMMENTS1').text if INDIVIDUAL.find('COMMENTS1') is not None else None
    Titulo=INDIVIDUAL.find('..//TITLE/VALUE').text if INDIVIDUAL.find('..//TITLE/VALUE') is not None else None
    Designacion=INDIVIDUAL.find('..//DESIGNATION/VALUE').text if INDIVIDUAL.find('..//DESIGNATION/VALUE') is not None else None
    Nacionalidad=INDIVIDUAL.find('..//NATIONALITY/VALUE').text if INDIVIDUAL.find('..//NATIONALITY/VALUE') is not None else None
    TipoLista2=INDIVIDUAL.find('..//LIST_TYPE/VALUE').text if INDIVIDUAL.find('..//LIST_TYPE/VALUE') is not None else None
```

Y se procede al insert de cada registro extraído

```
cursor.execute("INSERT INTO PersonasONU (DataID,Version,PrimerNombre,SegundoNombre,TipoLista,Referencia,FechaLista,Comentarios,Titulo,D  
Nacionalidad,TipoLista2,PaisDir,TipoFechaNac,AnioNac,CiudadNac,ProvinciaNac,PaisNac,SortKey,SortKeyLastMod)  
VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)", (DataID, Version, PrimerNombre, SegundoNombre, TipoLista, Referencia, FechaLista, Comentarios, Titulo, D  
Designacion, Nacionalidad, TipoLista2, PaisDir, TipoFechaNac, AnioNac, CiudadNac, ProvinciaNac, PaisNac, SortKey, SortKeyLastMod))
```

3. Se ha definido una función para cargar primero el archivo de la OFAC.

## Se hace la lectura del archivo XML y limpieza de las tablas

```
def cargarXMLofac(url):

    response = requests.get(url)

    # Verificar si la solicitud
    if response.status_code == 200:
        print("Archivo XML descargado correctamente.")
        xmlEstructura = xmlET.fromstring(response.content)
        namespaces = {'ns': 'https://sanctionslistservice.ofac.treas.gov/api/PublicationPreview/exports/XML'}

        conn=sqlite3.connect('listas0nuOfac.db')
        cursor=conn.cursor()

        cursor.execute("DELETE FROM ListaOFAC")
        cursor.execute("DELETE FROM ListaAkaOFAC")
        cursor.execute("DELETE FROM ListaAddressOFAC")
        conn.commit()
```

Luego se recorre el XML para recuperar cada registro.

```
for sdnEntry in xmlEstructura.findall('.//ns:sdnEntry', namespaces):
    Uid=sdnEntry.find('ns:uid', namespaces).text if sdnEntry.find('ns:uid', namespaces) is not None else None
    for aka in xmlEstructura.findall('.//ns:addressList//ns:address', namespaces):
        UidAddr=aka.find('ns:uid', namespaces).text if aka.find('ns:uid', namespaces) is not None else None
        Address=aka.find('ns:address1', namespaces).text if aka.find('ns:address1', namespaces) is not None else None
        City=aka.find('ns:city', namespaces).text if aka.find('ns:city', namespaces) is not None else None
        PostalCode=aka.find('ns:postalCode', namespaces).text if aka.find('ns:postalCode', namespaces) is not None else None
        Country=aka.find('ns:country', namespaces).text if aka.find('ns:country', namespaces) is not None else None

        cursor.execute("INSERT INTO ListaAddressOFAC (Uid,UidAddr,Address,City,PostalCode,Country) VALUES (?, ?, ?, ?, ?, ?)", \
            (Uid,UidAddr,Address,City,PostalCode,Country))

conn.commit()
```

4. Luego se define la función `main()` para la ejecución de las funciones anteriores.

```
def main():
    create_db()
    url1= 'https://scsanctions.un.org/resources/xml/en/consolidated.xml'
    cargarXMLONU(url1)
    url2= 'https://www.treasury.gov/ofac/downloads/sdn.xml'
    cargarXMLOFAC(url2)

if __name__ == '__main__':
    main()
```

5. Se aplico versionamiento con Git

Se preparo el repositorio local y se vinculó con repositorio remoto. Y se subieron los cambios.

<https://github.com/fabioladt/CargarListas.git>

```
C:\Users\Diego\Documents\Prueba_Tecnica_Fabi\ListasOnuOfac>git init
Initialized empty Git repository in C:/Users/Diego/Documents/Prueba_Tecnica_Fabi/ListasOnuOfac/.git/

C:\Users\Diego\Documents\Prueba_Tecnica_Fabi\ListasOnuOfac>git config --global user.name "Fabiola Duarte"

C:\Users\Diego\Documents\Prueba_Tecnica_Fabi\ListasOnuOfac>git config --global user.email "josselynfabiola@gmail.com"

C:\Users\Diego\Documents\Prueba_Tecnica_Fabi\ListasOnuOfac>git remote add origin https://github.com/fabioladt/CargarListas.git
```

```
C:\Users\Diego\Documents\Prueba_Tecnica_Fabi\ListasOnuOfac>git checkout -b master
Switched to a new branch 'master'

C:\Users\Diego\Documents\Prueba_Tecnica_Fabi\ListasOnuOfac>git add .

C:\Users\Diego\Documents\Prueba_Tecnica_Fabi\ListasOnuOfac>git commit -m "Cargando el archivo python de las listas ONU y OFAC"
[master (root-commit) 278a7d1] Cargando el archivo python de las listas ONU y OFAC
 4 files changed, 861727 insertions(+)
 create mode 100644 ListasOnuOfac.py
 create mode 100644 listasOnuOfac.db
 create mode 100644 listasOnuOfac.db-journal
 create mode 100644 sdn.xml

C:\Users\Diego\Documents\Prueba_Tecnica_Fabi\ListasOnuOfac>git push origin master
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 12 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 2.56 MiB | 1.08 MiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/fabioladt/CargarListas/pull/new/master
remote:
To https://github.com/fabioladt/CargarListas.git
 * [new branch]      master -> master

C:\Users\Diego\Documents\Prueba_Tecnica_Fabi\ListasOnuOfac>
```

The screenshot displays the GitHub interface for the repository **fabioladt / CargarListas**. The repository is marked as **Private**. A yellow banner indicates that the **master** branch has recent pushes 17 seconds ago, with a **Compare & pull request** button. Below this, the branch status shows **master** is **1 commit ahead of, 1 commit behind** the **main** branch. A file list shows three files: **ListasOnuOfac.py**, **listasOnuOfac.db**, and **listasOnuOfac.db-journal**, all with the status **Cargando el archivo**. A **Code** dropdown menu is open, showing options to **Clone** the repository using **HTTPS**, **SSH**, or **GitHub CLI**. The **HTTPS** option is selected, showing the URL **https://github.com/fabioladt/CargarListas.git** with a copy icon. Other options include **Clone using the web URL** and **Open with GitHub Desktop**.

6. Resultado de la extracción.

## Archivo ONU.

	DataID	Version	PrimerNombre	SegundoNombre	TipoLista	Referencia	FechaLista	Comentarios
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	110447	1	MUHAMMAD	TAHER	Taliban	TAi.005	2001-02-23	Belongs to Andar tribe. Review ...
2	110554	1	ABDUL LATIF	MANSUR	Taliban	TAi.007	2001-01-31	Taliban Shadow Governor for Logar ...
3	110578	1	ABDUL KABIR	MOHAMMAD JAN	Taliban	TAi.003	2001-01-25	Active in terrorist operations in ...
4	110579	1	MOHAMMED	OMAR	Taliban	TAi.004	2000-04-12	Father's name is Ghulam Nabi, also ...
5	110580	1	SAYYED MOHAMMED	HAQQANI	Taliban	TAi.006	2001-01-31	Graduate of the Haqqaniya madrasa in...
6	110896	1	SHAMS	UR-RAHMAN	Taliban	TAi.008	2001-02-23	Believed to be in Afghanistan/...
7	110897	1	MOHAMMAD NAIM	BARICH	Taliban	TAi.013	2001-02-23	Member of the Taliban Military ...
8	110899	1	FAZL MOHAMMAD	MAZLOOM	Taliban	TAi.023	2001-02-23	Review pursuant to Security Council ..
9	110900	1	SAID AHMED	SHAHIDKHEL	Taliban	TAi.028	2001-02-23	In July 2003 he was in custody in ...
10	110901	1	MOHAMMAD	AHMADI	Taliban	TAi.031	2001-02-23	Believed to be in Afghanistan/...
11	110912	1	ABDUL RAHMAN	ZAHED	Taliban	TAi.033	2001-01-25	Believed to be in Afghanistan/...

	DataId	FechaActualizacion
	Filter	Filter
1	110447	2003-09-03
2	110554	2003-09-03
3	110578	2003-09-03
4	110579	2003-09-03
5	110580	2003-09-03

	DataId	Calidad	Alias
	Filter	Filter	Filter
1	110447	Good	Mohammad Taher Anwari
2	110554	Good	Abdul Latif Mansoor
3	110578	Good	A. Kabir
4	110579	NULL	NULL
5	110580	Good	Sayyed Mohammad Haqqani
6	110896	Good	Shamsurrahman
7	110897	Good	Mullah Naeem Barech
8	110899	Good	Molah Fazl

**Archivo OFAC.**

	Uid	lastName	sdnType	programList
	Filter	Filter	Filter	Filter
1	36	AEROCARIBBEAN AIRLINES	Entity	CUBA
2	173	ANGLO-CARIBBEAN CO., LTD.	Entity	CUBA
3	306	BANCO NACIONAL DE CUBA	Entity	CUBA
4	424	BOUTIQUE LA MAISON	Entity	CUBA
5	475	CASA DE CUBA	Entity	CUBA
6	480	CECOEX, S.A.	Entity	CUBA
7	535	CIMEX	Entity	CUBA
8	536	CIMEX IBERICA	Entity	CUBA

	Uid	UidAka	Type	Category	lastName
	Fil...	Filter	Filter	Filter	Filter
1	36	12	a.k.a.	strong	AERO-CARIBBEAN
2	36	57	a.k.a.	strong	AVIA IMPORT
3	36	219	a.k.a.	weak	BNC
4	36	220	a.k.a.	strong	NATIONAL BANK OF CUBA
5	36	471	a.k.a.	strong	COIBA
6	36	475	a.k.a.	strong	CRYMSA
7	36	477	a.k.a.	strong	COPROVA
8	36	478	a.k.a.	strong	COPROVA SARL

	Uid	UidAddr	Address	City	PostalCode	Country
	Filter...	Filter	Filter	Filter	Filter	Filter
1	36	25	NULL	Havana	NULL	Cuba
2	36	129	Ibex House, The Minories	London	EC3N 1DY	Unit...
3	36	199	Zweierstrasse 35	Zurich	CH-8022	Swit...
4	36	200	Avenida de Concha Espina 8	Madrid	E-28036	Spain
5	36	201	Dai-Ichi Bldg. 6th Floor, 10-2 Nihombashi, 2-...	Tokyo	103	Japan
6	36	202	Federico Boyd Avenue & 51 Street	Panama City	NULL	Pana...
7	36	247	42 Via Brasil	Panama City	NULL	Pana...
8	36	271	NULL	NULL	NULL	Spain
9	36	272	NULL	NULL	NULL	Mexi...
10	36	276	NULL	Panama City	NULL	Pana...
11	36	317	Emerson No. 148 Piso 7	Mexico, D.F.	11570	Mexi...
12	36	318	NULL	NULL	NULL	Spain
13	36	319	NULL	NULL	NULL	Pana...