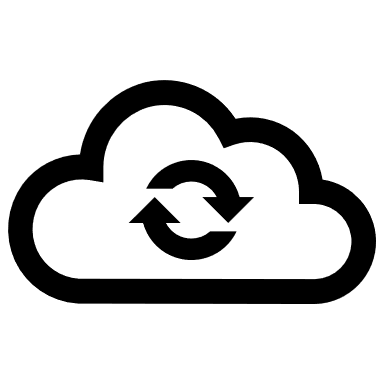
**Generalidades:** Ejecución del proyecto prueba para la vacante de Ingeniero de datos del Centro de Excelencia de Operación, generalidades respecto al código propuesto, requerimientos, ejecución del modelo, API, Power BI, y links tanto del repositorio en GIT como del video en YouTube.

**Control de versión**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Versión** | **Fecha de Modificación** DD/MM/AAAA | **Acción**  **C: Creación**  **M: Modificación** | **Nombre del responsable del cambio** | **Descripción breve del cambio** | **Aprobado Por** |
| 1 | 22/12/2023 | C | Harold Barrantes | Creación | Harold Barrantes |

1. **Requisitos para la ejecución del modelo** 
   * + Licencia LZ (PROCESADOR)
     + Python 3.9.7
     + Permiso External Table (sirve para escribir tablas en zona de procesos)
     + Cacerts para conexión lz (archivo en c)
     + PIP (archivo en el usuario)
     + Visual Studio Code
2. **Arquitectura del modelo**

********

**4 Excel Procesamiento Python Landing Zone LZ Excel**

El modelo está basado en Base calendarizable 2.0 que por Gobierno de información es la última versión disponible y recomendada por Banco, se recogen los insumos con Python, a través de Spark con Programación Orientada a Objetos se llevan a la LZ se procesan en SQL y de la misma manera se utilizan clases para almacenar el resultado en el dispositivo local, además, se gestiona un API.PY para hacer las pruebas de consulta desde la maquina en la web y se desarrolla el tablero en PBI con la situación de negocio planteada.

1. **Excel y Landing zone LZ fuentes:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tipo** | **Fuente** | **Información** | **Tabla y zona** |
| Excel | Insumo (Input) | Obligaciones\_clientes.xlsx | proceso.CDEOP\_prueba\_sql1 |
| Excel | Insumo (Input) | tasas\_productos.xlsx | proceso.CDEOP\_prueba\_sql2 |
| Excel | Insumo Output | Resultado\_1.xlsx | proceso.CDEOP\_prueba\_sql1 |
| Excel | Insumo Output | Resultado\_2.xlsx | proceso.CDEOP\_prueba\_sql2 |

**Librerías:**

Las librerías se instalan desde el archivo setup.cfg en el entorno virtual de forma automática (leer el ARCHIVO README), de las cuales las usadas en el proyecto son:

1. pyodbc==4.0.27

2. orquestador2>=1.2.2

3.Pandas (para el manejo de los Dataframe)

4.Openpyxl (para trabajar cómodamente con la escritura y lectura de archivos xlsx)

5.Flask (para el uso del API)

Para entender más a detalle el funcionamiento de la rutina y en caso de ser necesario realizar futuros ajustes o intervenciones al proyecto por parte del usuario, se deja la explicación de los principales parámetros, dentro del proyecto se define el archivo “config.json” que contiene los parámetros que se utilizarán en las consultas:

dsn: El sistema de nombres de dominio. Ej (IMPALA\_PROD)

username: Nombre del usuario para el uso de la LZ.

1. **Procesamiento**

**ETL Parte 1 y 2**

* + - Se verifican los requisitos mínimos
    - Se instala el proyecto en el equipo de quien va a ejecutar el modelo (**usar el Readme**)
    - Activar el entrono virtual
    - Ejecutar el modelo Python para la ETL desde la terminal de Visual Studio Code (ejecución.py)
    - Ingresar contraseña del usuario con permisos LZ
    - Revisar en la carpeta “Output” el resultado de la ejecución

**API Parte 3**

* Conexión a internet
* Requisitos mínimos (Leer el Readme)
* Activar el entorno virtual
* Ejecutar el modelo Python para la ETL desde la terminal de Visual Studio Code (API.py)

1. **Arquitectura**

La arquitectura del proyecto se divide en carpetas y archivos que permiten realizar la ejecución del proyecto.

* **venv:** Carpeta que almacena el entorno virtual dispuesto para este proyecto.
* **Setup.cfg:** Archivo que permite gestionar los parámetros de instalación de librerías y requisitos del orquestador 2.0
* **inputs:** Carpeta que almacena los archivos insumo, los cuales son requeridos para la ejecución del proyecto.
* **outputs:** Carpeta que almacena los archivos salida o resultados de la ejecución del proyecto.
* **\vedd-prueba-cdeop\src\vedd\_prueba\_cdeop\static\sql\ExtractTransformLoad:** Carpeta que contiene los archivos en formato SQL, necesarios para el procesamiento de la información necesaria para obtener los resultados esperados por el proyecto.
* **logs:** Carpeta en donde quedan los archivos que dejan el respaldo y hacen seguimiento a las ejecuciones de la rutina, en donde identifica si se ejecutó correctamente o tuvo problemas.
* **\vedd-prueba-cdeop\src\vedd\_prueba\_cdeop\static\Config.json:** Archivo “.json” con la definición de parámetros como zonas de ejecución en la LZ, las clases para subir y bajar archivos con Spark.
* **vedd-prueba-cdeop\src\vedd\_prueba\_cdeop\etl.py:** Código de Python que contiene los steps de la prueba, donde por medio de lenguaje Python se procesa la información de los insumos hasta los archivos resultados.

**ExtractTransformLoad(Step):** Parámetros de ejecución del modelo. Cargue de fechas temporales y ejecución de los archivos almacenados en la carpeta que contiene los SQL.

**Subir\_excel(Step):** Clase que permite cargar en Dataframe los insumos hacer alguna transformación, invocar los parámetros y llevar los datos a la LZ.

**GuardarCopiasExcel(Step):** Clase que a través de Helper descarga en Dataframe los resultados y permite llevarlos en formato xlsx con pandas a una carpeta local.

* **API.py** : Python con las funciones necesaria a través de la librería flask para el funcionamiento del API, además, llama en Dataframe los resultados de la ejecución del ETL de la parte 1 y 2

1. **Detalles del procesamiento**

ETL Python (ejecución.py):

En Python (etl.py) se establecen 3 steps o pasos que permiten cargar los archivos de Excel con algunos ajustes a la LZ, ejecutar los query y generar el resultado de la siguiente forma:

1 step: Subir archivos lz

Este paso permite subir los 2 archivos de Excel a la LZ, el procesamiento es el siguiente:

1. Archivo de **Obligaciones\_clientes:**

* Sube el archivo a Python
* Transforma los formatos de algunos campos para estandarizarlos
* Carga de dataframe a la LZ

1. Archivo **tasas\_productos:**

* Sube el archivo a Python
* Transforma los formatos de algunos campos para estandarizarlos
* Carga de resultado a la LZ

2 step: Ejecutar SQL asociados al proyecto

Desde SQL y a partir de las ejecuciones anteriores se establecen 09 querys que se describen a continuación:

* **001\_consulta1.sql**: Realiza la parte 1 del proyecto incluido el calculo de la Tasa Efectiva TE y deja el resultado del punto 3 con sus requerimientos.
* **002\_consulta2.sql**: Realiza la parte 1 del proyecto incluido el cálculo de la Tasa Efectiva TE y deja el resultado del punto 4 con sus requerimientos.

3 Step descargar el resultado

El tercer paso es descargar el resultado en la ruta del proyecto (**\vedd-prueba-cdeop\src\vedd\_prueba\_cdeop\output**), estos son dos archivos de Excel con la información requerida de obligaciones y tasas.

API (ejecución Python API.py):

A través de Flask en la terminal al realizar el llamado del Python crea el endpoint en sonse se puede dar ctrl + clic o copiar y pegar en el navegador de preferencia el endpoint y adicionar “/exercise1/<num\_documento>” o “/exercise2/<num\_documento>”, según si se desea observar todos los productos por cliente, o únicamente el valor agregado por cliente respectivamente.

(Ejemplo:

<http://127.0.0.1:5000/exercise1/1277982669>

[http://127.0.0.1:5000/exercise2/ 1277982669](http://127.0.0.1:5000/exercise2/%201277982669))

Power BI

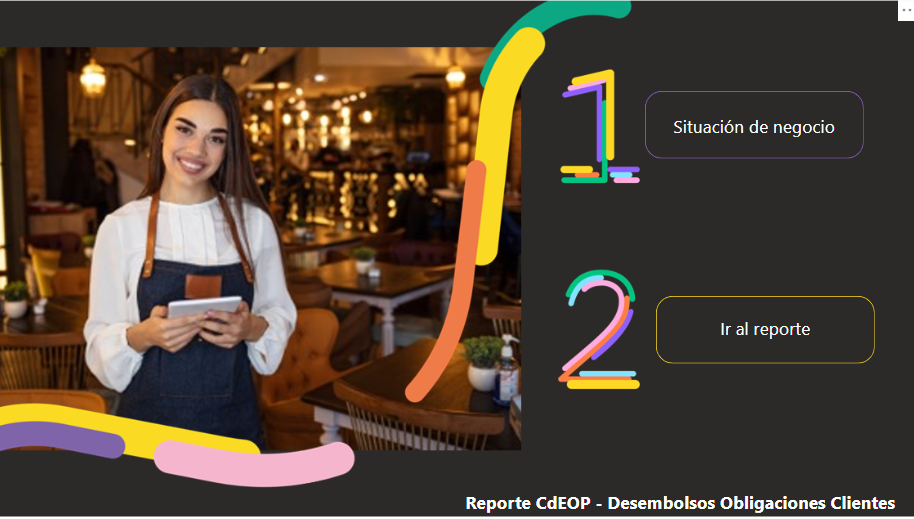
Se requiere tener instalado Power bi desktop y abrir el archivo del repositorio

**Reporte desembolsos - Obligaciones clientes.pbix**

**En él se encontrarán dos páginas:**

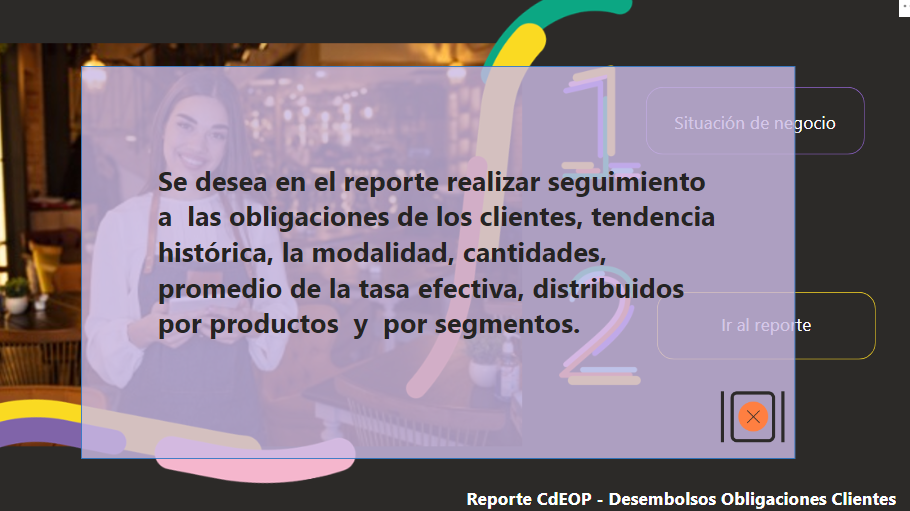
**Home**

El Home es la bienvenida al reporte en la primera visual se observar el nombre del reporte y dos marcadores a través de botones que se activan dando CTRL + CLIC por la versión locar de Power BI, la vista es así:



**Situación de negocio:**

Al dar CTRL + CLIC en el botos azul situación de negocio abrirá una ventana emergente desarrollada a través de marcadores de Power BI con la explicación de lo que se desea evaluar, su vista es la siguiente:



**Ir al reporte**

Ir al reporte permite dirigirnos a la segunda pagina del informe llamada “Obligaciones Clientes”, donde se pretende hacer un análisis anual del comportamiento d ellos desembolsos de los clientes por modalidad, tipo de producto, segmento, filtrado por fechas como año y mes y con algunas connotaciones diferenciales como el valor mínimo y máximo de los desembolsos y las TE que corresponden a los filtros que se realicen el en mismo, la vista es la siguiente:



Para visualizar la explicación de este reporte se puede ingresar al link de YouTube:

<https://youtu.be/hgazM1Q4TgQ?si=TtyKyGCtrLUmIDia>

**REPOSITORIO GIT**

Los archivos como insumos y resultados, documentación, README, POWER BI, necesarios para este proyecto los pueden encontrar en el siguiente link:

<https://github.com/haroldbarrantes/Prueba_ID_CdEOP_1>

NOTA:

La carpeta con todo el desarrollo no tendrá el entorno virtual instalado para la ejecución, se recomienda usar el README para instalar los componentes y que el modelo funcione, la vista es la siguiente:

