**Prueba técnica para Ingeniería de Datos Jikkosoft**

**Documentación**

**Respuesta al reto**

Iniciare por presentar el diagrama para el pipeline propuesto, donde utilicé Python y MySQL.

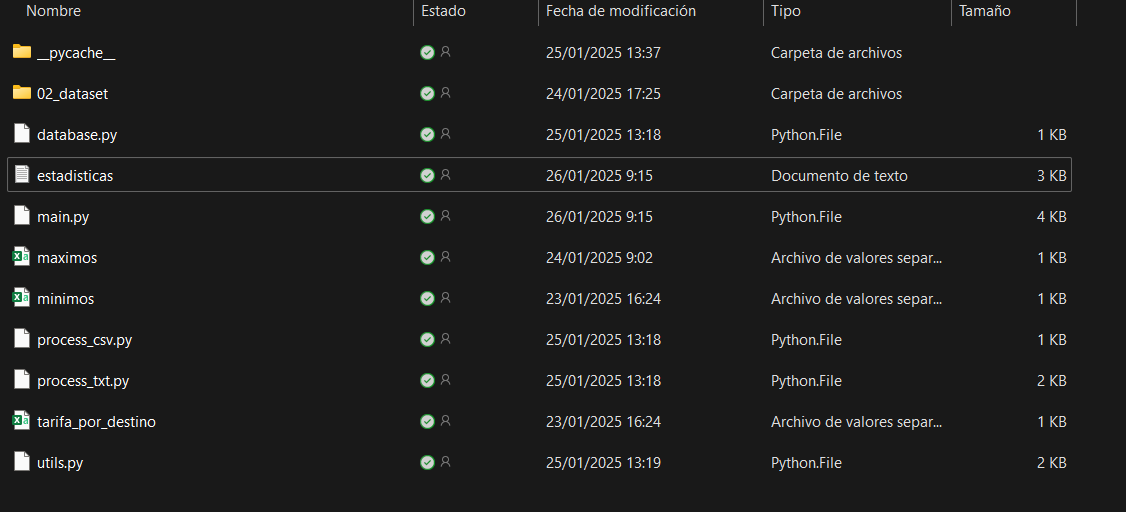
Imagen que contiene Diagrama

Descripción generada automáticamente

**Requerimientos:**

1. Descarga la carpeta comprimida que contiene los datos y déjalos en una carpeta exclusiva para este reto.

La carpeta es descargada y en ella misma montamos todo el proyecto, organizando los .py en orden de ejecución. Conservando los .txt en una carpeta propia, y utilizando los .csv en el mismo proyecto.



1. Construye un Pipeline que sea capaz de:
2. Cargar todos los archivos .TXT (El pipeline no debe contener todo el conjunto de datos, es decir, los cinco archivos al mismo tiempo en memoria en cualquier momento).

Con la función *read\_txt\_in\_batches* se asegura la carga de los archivos .txt.

1. Cada Micro Batch debe contener como máximo 1000 registros.

Establecido el máximo en 1000 registros, el *batch\_size* se establece en lotes de 500 registros, lo anterior dado que al analizar dataset-1 este contiene 500 registros, y los dataset en conjunto conforman 18500 registros, por lo que se diseña el pipeline para cargarlos en lotes de 500 registros.

La función *fill\_missing\_values* asegura que las *tarifas\_por\_destino.csv* que no contienen un estrato como, comercial, industrial, especial y otros, tarifados al 1.7% les asignen el estrato 7, y para el caso de oficial el estrato 4 ya que el residencial estrato 4, tarifa al 1%. En resumen, se evitan los null, o datos vacíos, en este diseño, y asegura que las columnas tengan un tipo de dato definido.

Texto

Descripción generada automáticamente

1. Almacena los datos de los archivos .TXT en una base de datos de tu elección (ejemplo: PostgreSQL, MySQL, MongoDB, etc). El diseño de esta base de datos dependerá de ti, crea la tabla o tablas que creas necesarias con el esquema que creas es adecuado, pero ten presente que todos los .TXT deben ir en la misma base de datos.

El pipeline diseñado almacena los datos en una base denominada Jikkosoft, en la tabla consumos, puede apreciar el entorno workbench de MySQL, con los datos cargados, 18500 registros.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

1. Calcular la tasa de impuestos al consumo teniendo en cuenta:
2. Tarifa sobre el consumo para calcular la tasa: tarifa\_por\_destino.csv
3. Cálculo de la tasa: Tarifa sobre consumo \* Consumo = Tasa Calculada
4. Valores mínimos aceptados: minimos.csv
5. Valores máximos aceptados: maximos.csv

Los 4 literales se realizan en la función *process\_and\_insert\_batches,* en el archivo principal *main.py*, previo procesamiento de los .csv en *procces\_csv.py* en la siguiente línea.

Texto

Descripción generada automáticamente

1. La Tasa Calculada por registro debe modificarse según las reglas de las tablas de mínimos y máximos.

Tomando una muestra, los siguientes registros son modificados con las reglas de mínimos, donde residencial estrato 1 tiene una tarifa de consumo de 0%, la tasa calculada operando da como resultado cero por tanto para 2024 residencial estrato 1 se establece tasa\_calculada = mínimo = 2353.25, para 2023 en 2120.6, en máximos no se presenta ningún caso.

Interfaz de usuario gráfica, Tabla

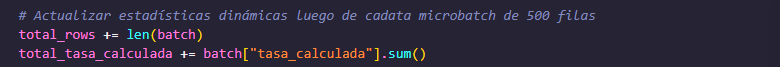
Descripción generada automáticamente

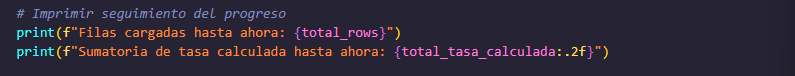
1. A medida que los datos son cargados, realiza un seguimiento de:
2. Recuento (COUNT) de filas cargadas a la base de datos.
3. Sumatoria (SUM) de la tasa calculada.

**NOTA:** Se espera que, en la ejecución de este pipeline, al menos después de que se cargue cada .TXT (pero idealmente después de la inserción de cada fila), estas estadísticas se actualicen para reflejar los nuevos datos. Las actualizaciones del Pipeline NO deben tocar los datos ya cargados, es decir, hacer una consulta para traer los valores esperados para cada actualización no es una buena solución al problema.

Es posible observar estos dos parámetros en la terminal de vscode, al ejecutar el pipeline. Adicional la carga por lotes de 500 registros asegura que las estadísticas de filas y tasa calculada se actualicen. Como se realiza con variables acumuladoras en ningún momento el diseño realiza una consulta a la base.







Texto

Descripción generada automáticamente

**Comprobante de resultados:**

1. Imprime el valor actual de la tasa en ejecución.
2. Realiza una consulta en la base de datos de: recuento total de las, valor promedio, sumatoria, valor mínimo y valor máximo para el campo calculado “Tasa”.

Para estos dos puntos se observa en la terminal de vscode, la tasa de ejecución en los lotes de 500 registros que no hacen llamado alguno a la base y las estadísticas solicitadas estas si se obtienen mediante consulta a la base mediante el llamado de la función *get\_statistics* declarada en el archivo *utils.py* e importado al *main.py.* Usando la conexión a la base *Jikkosoft* en MySQL. Es importante concluir que, la sumatoria de tasa calculada hasta ahora: 2669191325.71 obtenida de las variables en el pipeline es igual a la sumatoria de tasa calculada: 2669191325.71 obtenida de una consulta a la base de datos, indicando que se cumple la comprobación.

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

**Algunas reglas y consideraciones del reto:**

* Puedes utilizar cualquier Framework o librería que desees.

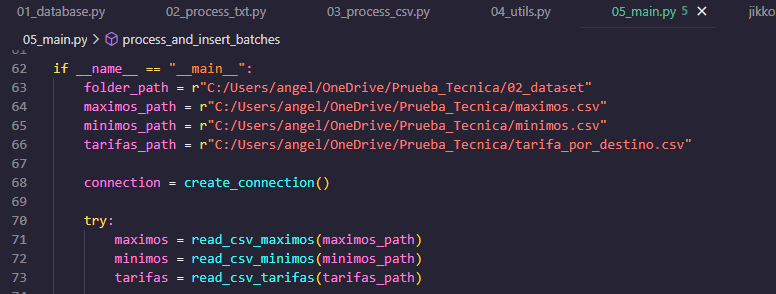
Se utilizo *mysql.connector – pandas – os*, en Python.

* Puedes utilizar cualquier base de datos que desees, bien sea SQL o NoSQL, lo importante es que muestres cómo te conectas a ella, cómo poblas la(s) tabla(s) y como realizas las consultas.

Se utilizo MySQL, se define la función *create\_connection* en el archivo *database.py,* esta función se utiliza en la función *get-statistics* definida en el archivo *utils.py,* donde realizamos la consulta de estadísticas directamente a la base *Jikkosoft*, de igual manera se usa la conexión (connection) en el archivo principal *main.py* para poblar la base de datos. Es importante aclarar que desde vscode se realiza el query para crear la base de datos en MySQL.

Texto

Descripción generada automáticamente



Texto

Descripción generada automáticamente

* Puedes hacer uso de alguna interfaz gráfica para administrar/manipular tu base de datos (ejemplo PgAdmin), o puedes hacer uso de línea de comandos.

Se utilizo workbench de MySQL, para confirmar la carga de la base.

Interfaz de usuario gráfica, Texto, Tabla

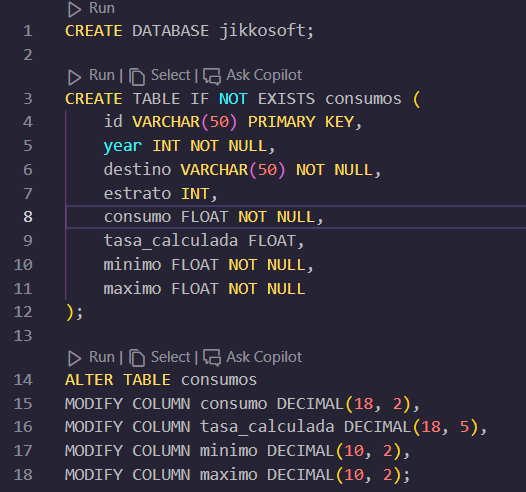
Descripción generada automáticamente con confianza media

* Puedes usar cualquier código existente que tengas a disposición.
* No hay una forma definida de resolver esta tarea, queremos ver la forma en la que piensas para resolver un problema así.
* Las estadísticas se pueden almacenar de la forma que desees: en base de datos, en memoria, en un archivo. Se almacenan en un estadisticas.txt
* El objetivo es una solución funcional, sin embargo, le damos alto valor al rendimiento.

**[](https://1drv.ms/f/c/e38df043e5ae12f9/EgYVnuLH4JVMqEzmZgRb218BvhglYD1NgMKaG3C768RloQ?e=vpbhNC)Acceda a losa archivos dando clic en el icono de carpeta.**

**Orden de ejecución.**

1. Crear base de datos jikkososft, crear tabla consumos, ajuste a decimales.

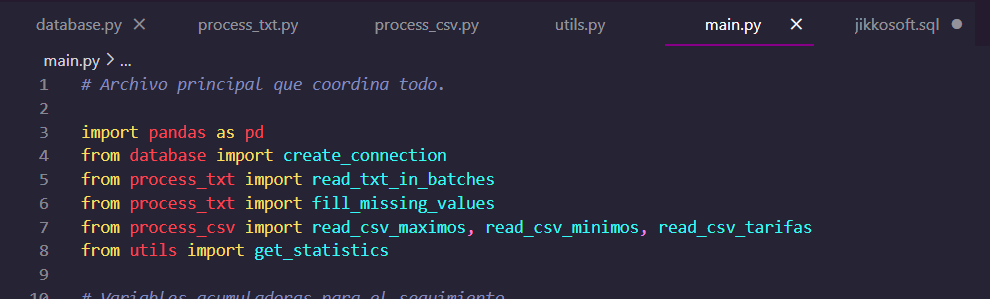


1. Modifique sus credenciales, en el archivo *database.py.*

Texto

Descripción generada automáticamente

1. Ejecute *main.py*



1. Revise los resultados en la terminal de vscode.

Texto

Descripción generada automáticamente

Altamente agradecido, por la experiencia adquirida en esta prueba técnica.

Cordialmente.

Angel Vera

Ingeniero Electrónico – Magister en Transformación Digital