**Explicación de las decisiones tomadas en el desarrollo**

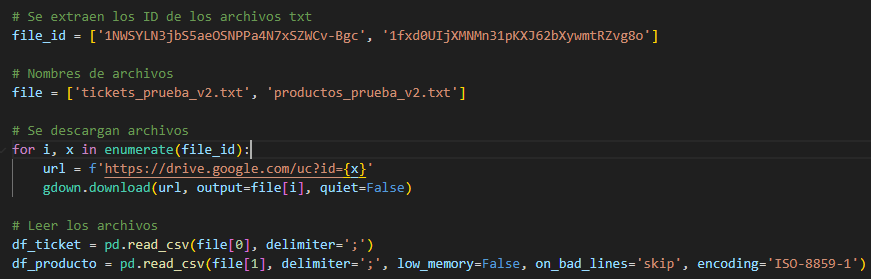
**Parte 1: Extracción de datos.**

Para la extracción de datos utilizo la librería gdown, esto descarga de manera local los archivos almacenados en el drive.

**Obs.** Intente hacerlo sin descargar de manera local con request y StringIO, pero una no podía por la verificación de virus y la otra no permitía por la codificación.

Luego esto lo llevo a los dataframe df\_ticket y df\_producto.

En el caso de df\_producto generaba errores en la codificación debido a la mezcla de datos y registros malos que rompían el formato. En caso de los registros malos que rompían la tabla aplique on\_bad\_lines que saltaría estas líneas y solo me permitió codificar con ISO-8859-1.



**Parte 2: Transformación y carga de datos**

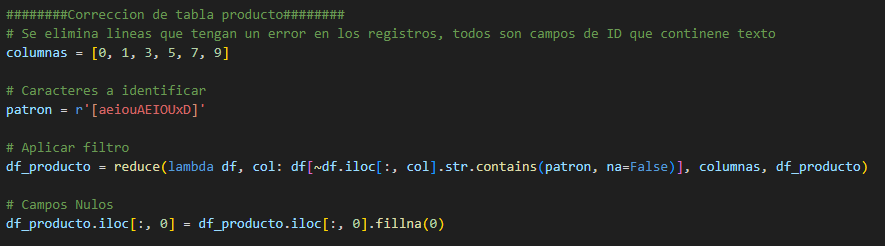
Luego realice una normalización de la tabla de productos.

Identifique las columnas que corresponden a ID que en este caso solo debe contener números e indique los caracteres que consideraría.

Para aplicar el filtro utilice la librería “importe functools import reduce” a modo de reducir el código (inicialmente había hecho una línea por columna).

La función toma las columnas mencionadas anteriormente y filtra esas filas con registros malos que contengan textos en los campos de ID.

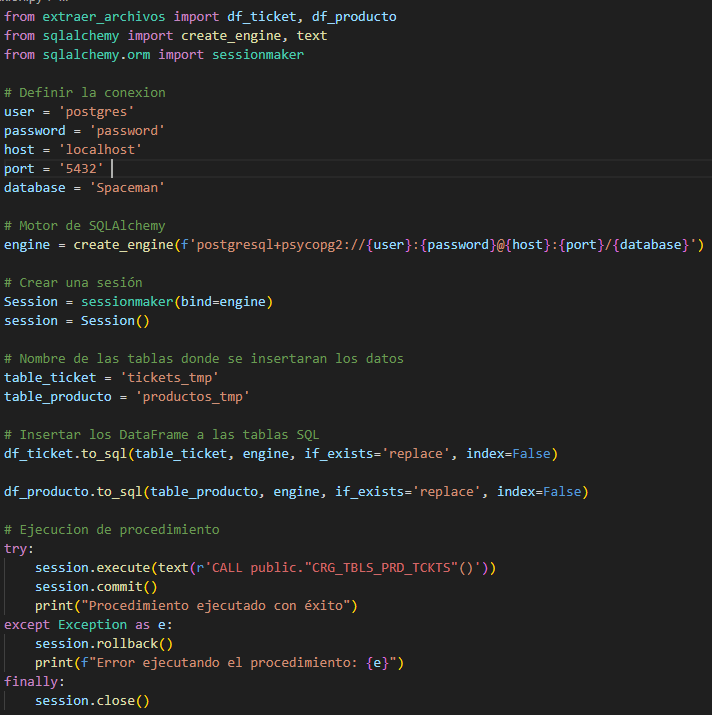
Seguí esta lógica a modo de no romper la estructura a utilizar en la tabla de SQL debido a que había líneas que contenían “;;” y también líneas incompletas con “\_x000D\_”.

También completo los campos NULL en la columna de “idcadena” para que no rompiera las consultas SQL.

Luego cree una librería para reemplazar todos los caracteres especiales que no pueden ser procesados por la codificación ISO-8859-1.

Para la conexión cree otro archive python para no mezclar el código.

Una vez creada la conexión con “sqlalchemy” creo o reemplazo las tablas “tickets\_tmp” y “productos\_tmp”, serán tablas temporales que utilizare antes de llevarlas a las tablas finales.

Y por último llamo a un procedimiento del postgresql utilizando “sqlalchemy.orm” que se encarga del volcado a la tabla final y borrado de las tablas temporales.

La idea al almacenar antes en una tabla temporal y luego a la tabla final es realizar las conversiones de campos necesarias y correcciones previas en la tabla como en caso de la Ñ que por la codificación realizada anteriormente se generó con un carácter que Python no identifica, pero si el SQL.

Luego tomo las tablas temporales corregidas y las inserto en la tabla final “productos” y “tickets”. También hice una verificación de que los registros a insertar no existiesen para evitar duplicaciones (en el caso de tickets lo comenté porque el proceso pesaba demasiado siendo 5 millones de registros).

Por último, se realiza el borrado de las tablas temporales.

