Análisis de datos Contratación Secop II

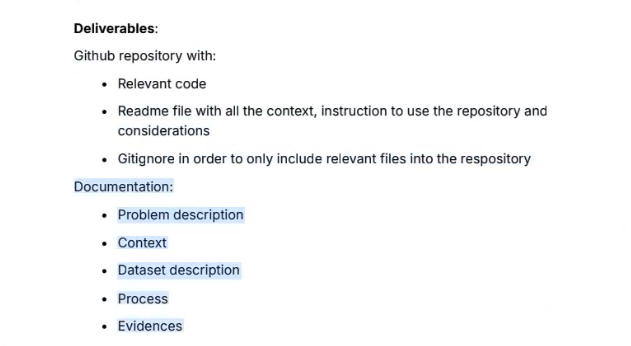
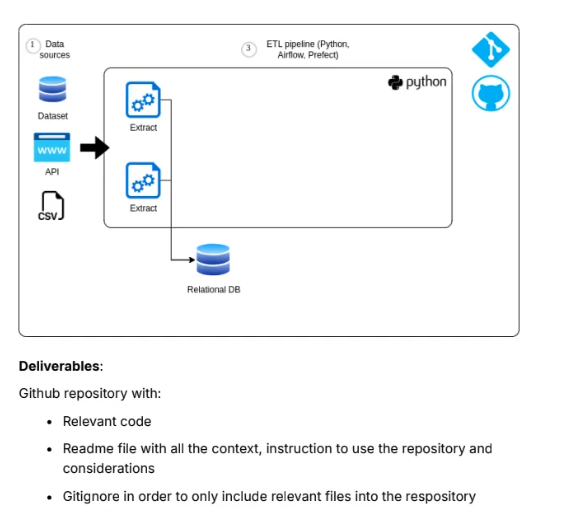
Proyecto de Curso ETL

Maestría en IA y BigData

Integrantes:

Jeicson Andrés Zúñiga Olmos

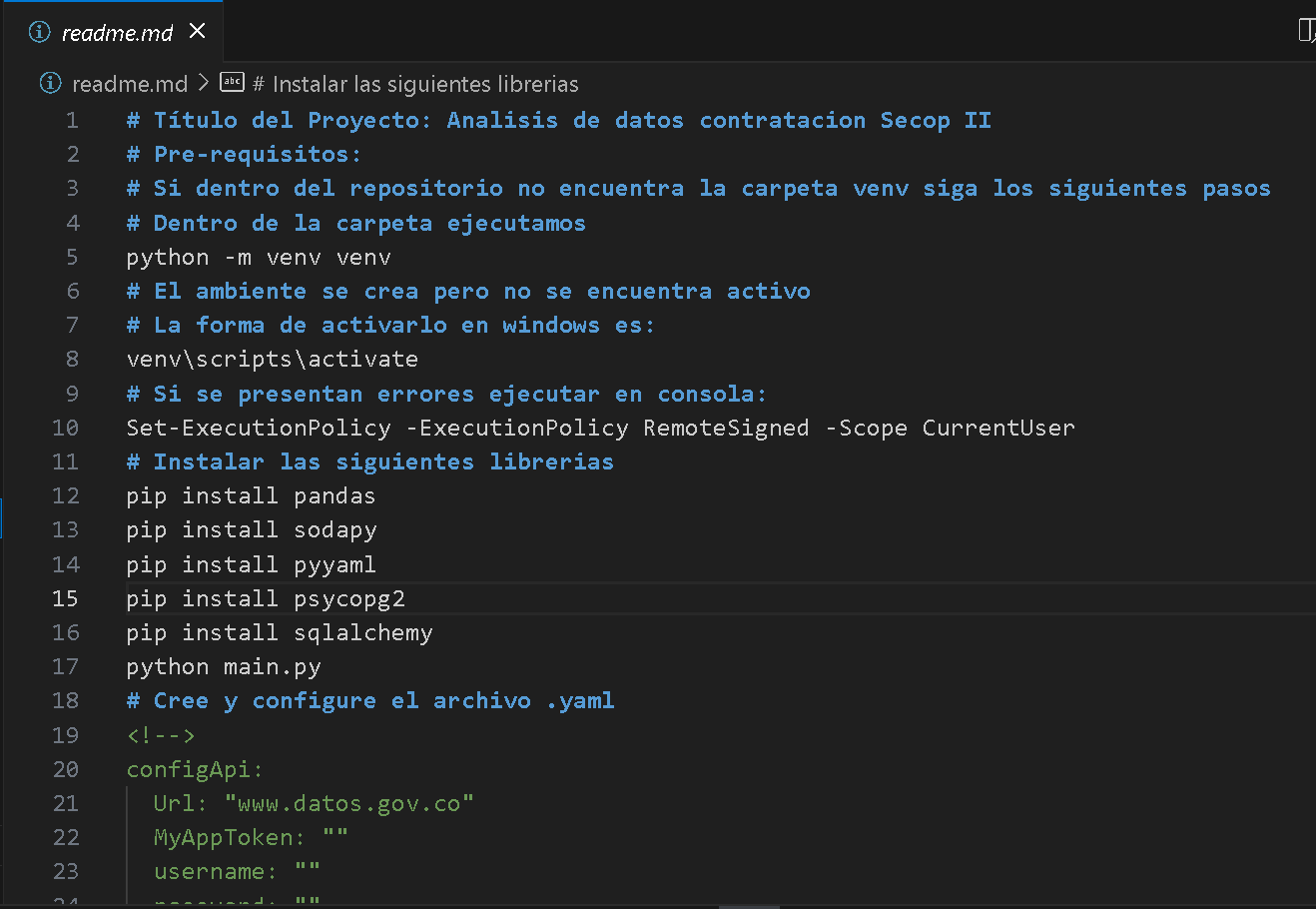
Carlos Fernando Escobar



Entregables:

Repositorio de GitHub con:

* Código relevante
* Archivo Léame con todo el contexto, instrucciones para usar el repositorio y consideraciones



* Git ignore para incluir solo archivos relevantes en el repositorio





Documentación:

* Descripción del problema

Se requiere consolidar gran parte de la información del proceso de contratación del Secop II contratos digitales con el objetivo de analizar los destinos del gasto y el sector donde aplica para determinar su comportamiento anual, semestral y mensual por entidades, departamentos y municipios.

* Contexto

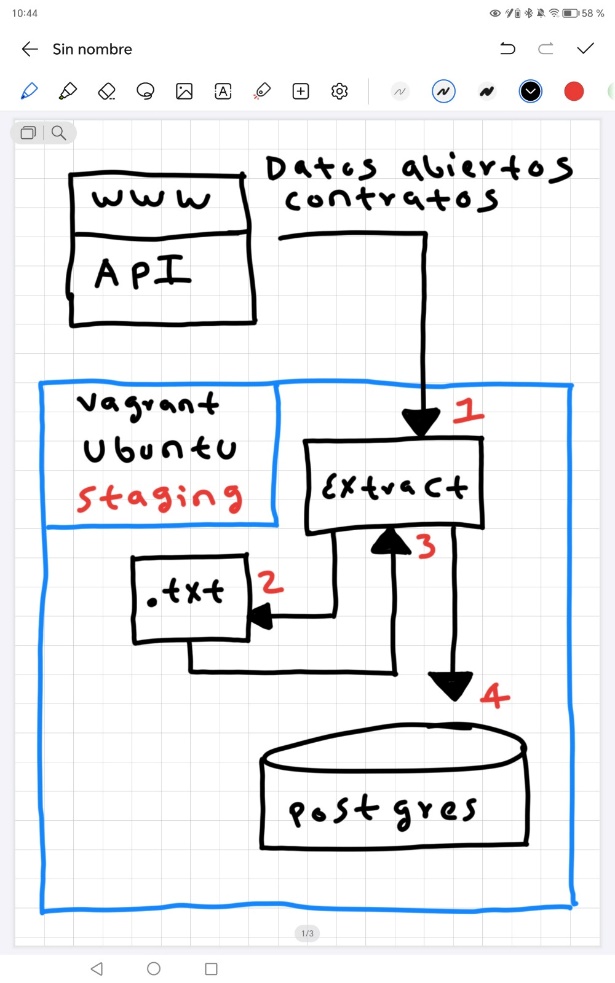
SECOP 2 (Sistema Electrónico de Contratación Pública 2) es una plataforma en línea implementada por el Gobierno de Colombia para gestionar los procesos de contratación pública de manera transparente, eficiente y accesible. Es una herramienta que facilita tanto a las entidades del Estado como a los proveedores la participación en los procesos de adquisición de bienes y servicios a nivel nacional.

* Descripción del conjunto de datos

1. Nombre Entidad: Nombre de la Entidad con la que se desarrolló el proceso de compra
2. Departamento: Departamento en el cual se registró la entidad del estado que publica el contrato
3. Ciudad: Ciudad de Colombia en el cual se registró la entidad del estado que publica el contrato
4. Orden: Orden entidad del estado que publica el contrato (Nacional, Territorial)
5. Sector: Sector de la economía de la entidad del estado que publica el contrato
6. Rama: Rama del Poder a la que corresponde la entidad del estado que publica el contrato
7. Estado Contrato: Estado del contrato, frente a su ejecución, firma o liquidación
8. Tipo de Contrato: Tipo de contrato de acuerdo a su marco jurídico
9. Modalidad de Contratación: Modalidad de contratación de acuerdo al modelo de selección
10. Documento Proveedor: Número de documento del proveedor adjudicado
11. Destino Gasto: Destino del gasto, a nivel presupuestal
12. Valor del Contrato: Valor total del contrato
13. Nacionalidad Representante Legal: Nacionalidad del representante legal de la empresa proveedora
14. Genero Representante Legal: Género del representante legal
15. Fecha de Firma: Fecha en que fue firmado digitalmente el contrato
16. Fecha de Inicio del Contrato: Fecha de inicio de las responsabilidades contractuales
17. Fecha de Fin del Contrato: Fecha de fin de las responsabilidades contractuales

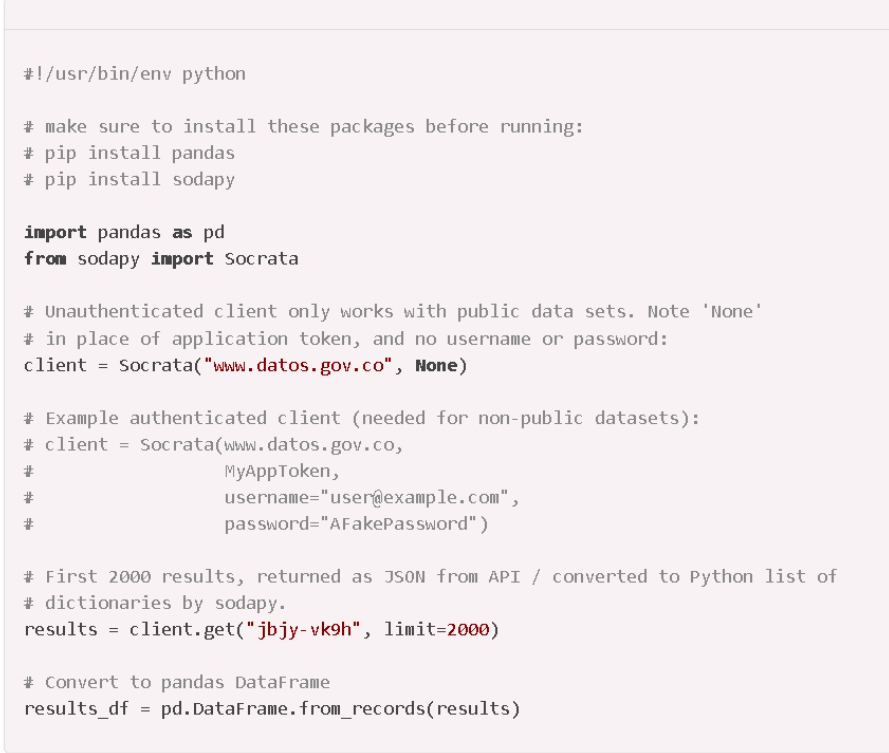
* Proceso

1. Se crea el acceso a la plataforma de datos abiertos de Colombia. https://www.datos.gov.co/
2. Se lee la documentación de la API y se crean el token de acceso para realizar la conexión por medio de Python.
3. Se crea un ambiente y se realizan las configuraciones necesarias para acceder a la información de la API por medio de Python.
4. Se consulta toda la información y se almacena en un archivo de texto de manera local en el área de Staging.
5. Se crea la base de datos etl\_proyecto\_curso para almacenar la información.
6. Se analizan las columnas proporcionadas por la API para crear la tabla que va consolidar la información.
7. Se crea la tabla apl1\_contratos\_digitales.
8. Se empieza el proceso de lectura de datos para ser almacenados en la base de datos.
9. Se encuentran errores durante la ejecución y se procede a crear un método de validar del formato json para evitar que el programa se detenga y continuar con el proceso.
10. Se inyectan todos los datos en la base de datos.

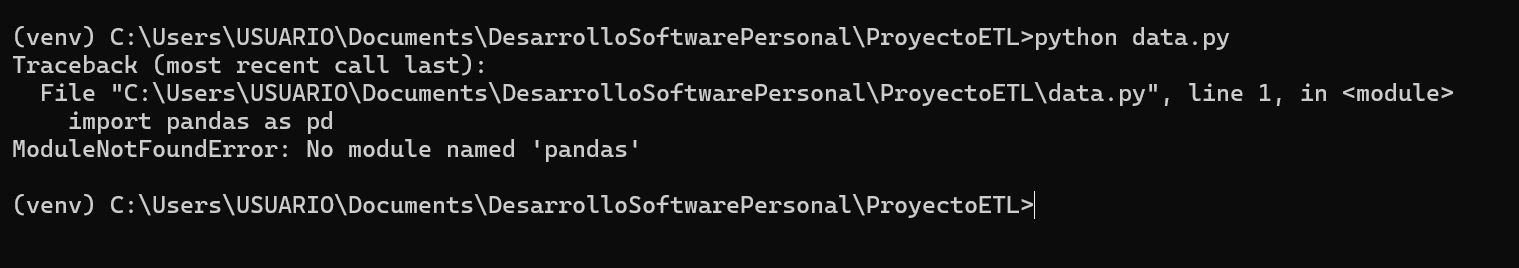


* Evidencias

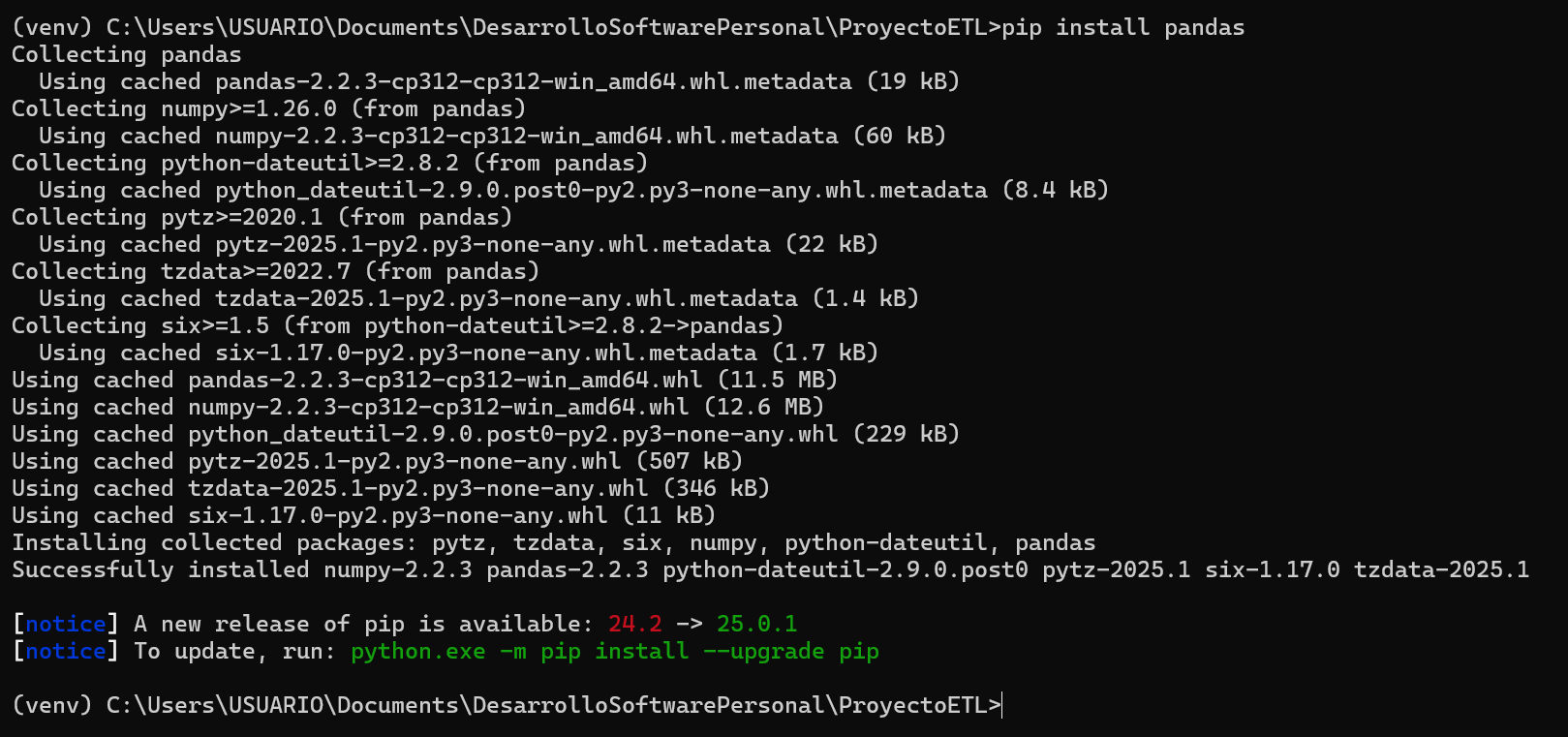
Se crean las credenciales de conexión para tener acceso al token que permite interactuar con la API de datos abiertos contratos electrónicos libremente.



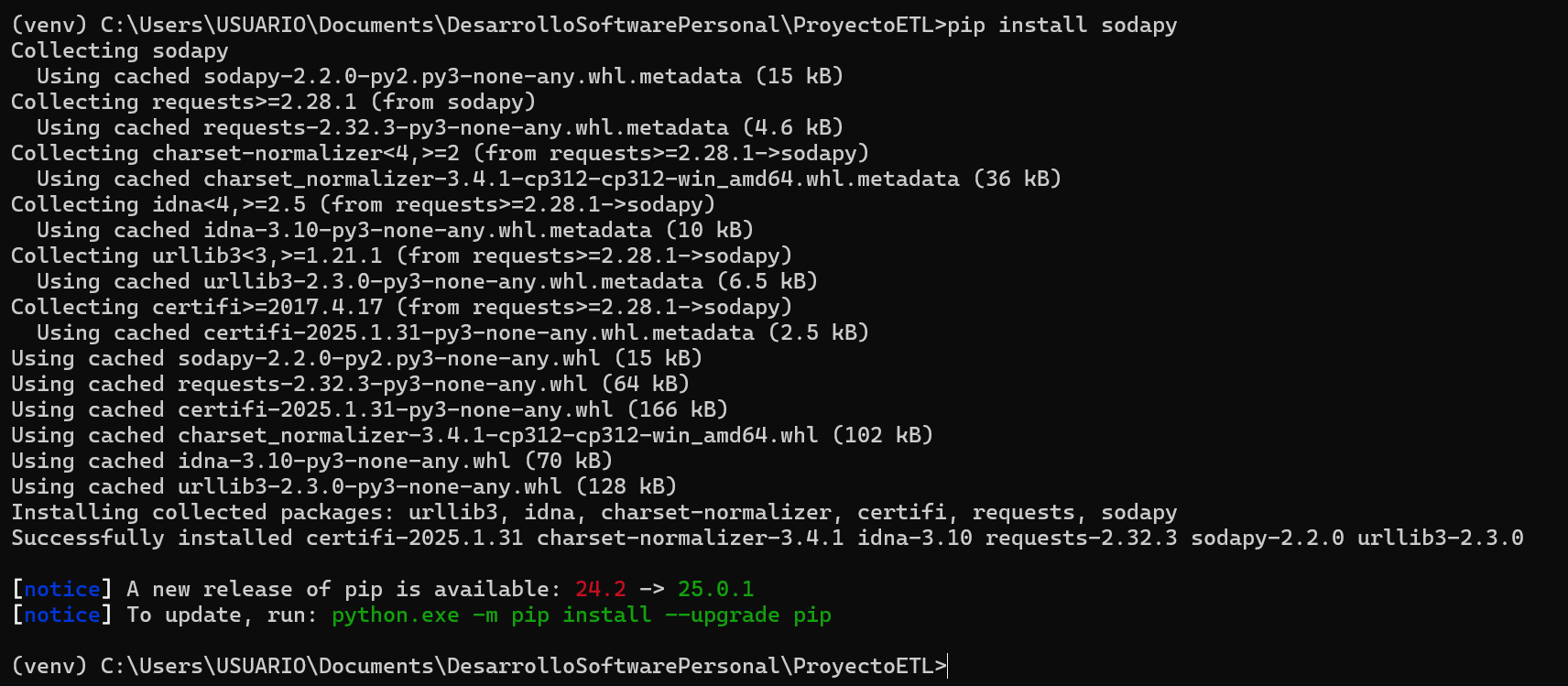
Se crea el ambiente para desarrollo.

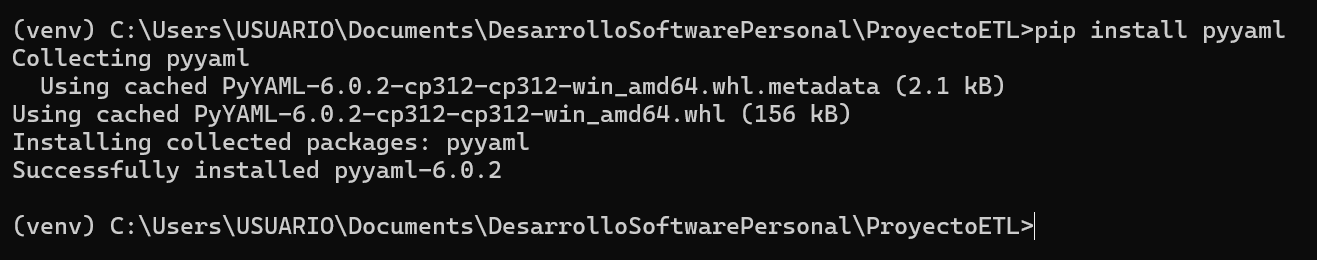


Se instala la librería de pandas sobre el ambiente.

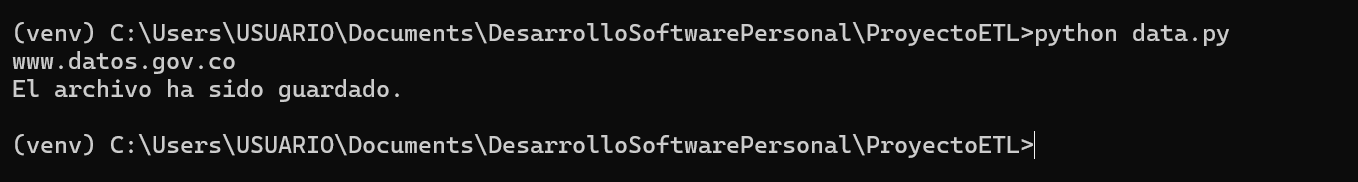


Se instala la librería de sodapy sobre el ambiente.





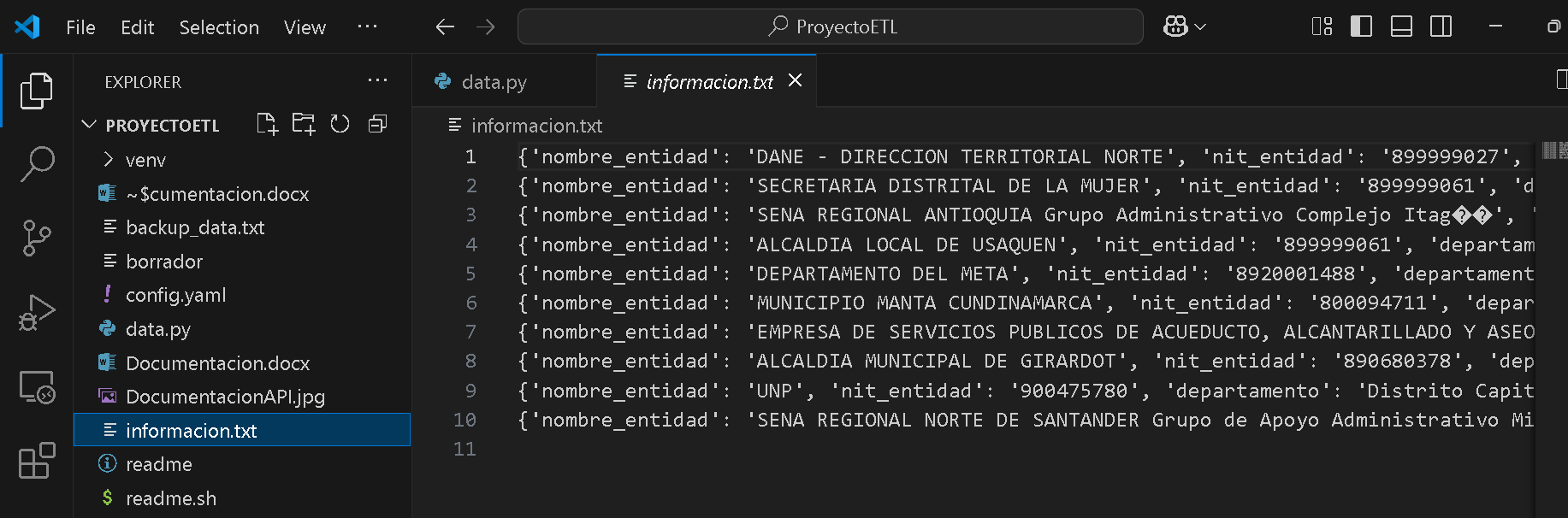
Se logra conectar con las credenciales de conexión a la api de datos abiertos del secop II



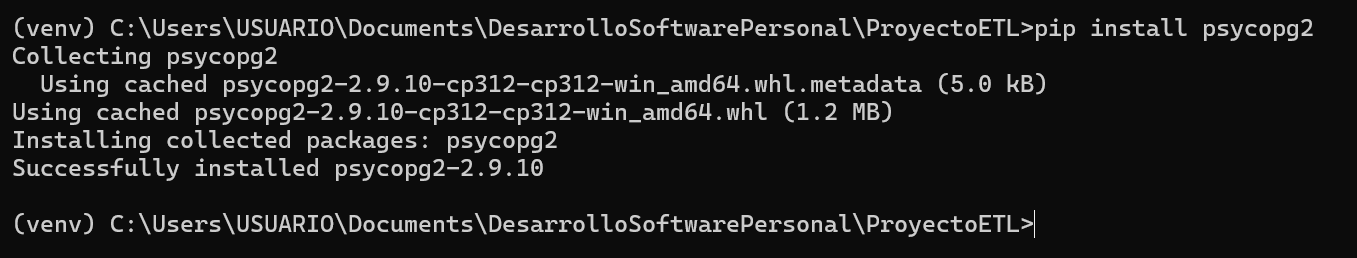
Se guarda la información realizando pruebas de 10 registros en un archivo llamado información.txt

Probando a través de la url publica que limita solamente a 1000 registros.

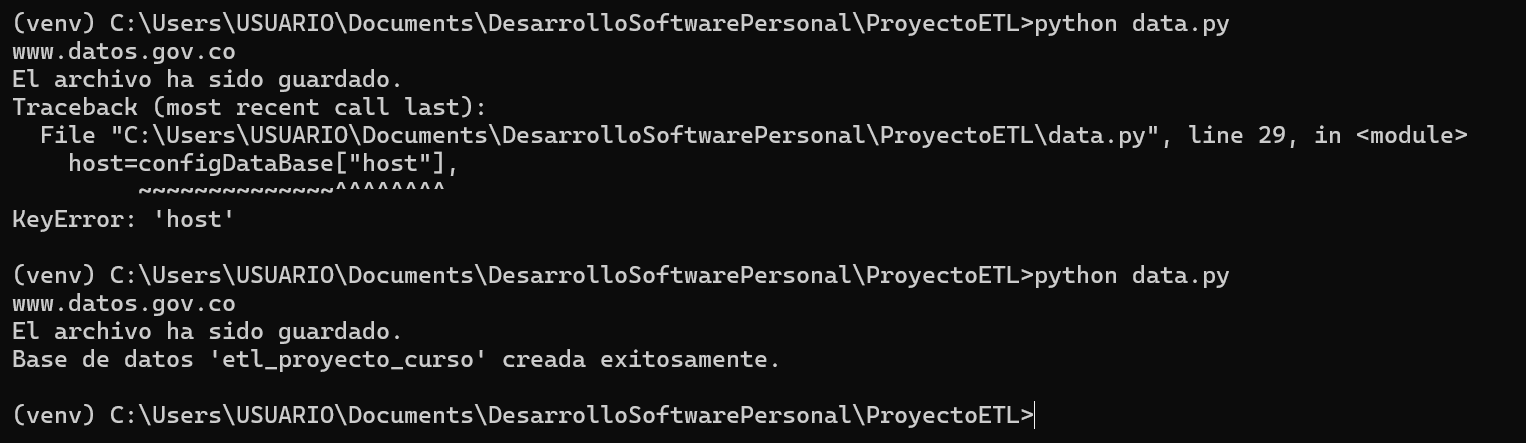
https://www.datos.gov.co/resource/jbjy-vk9h.json



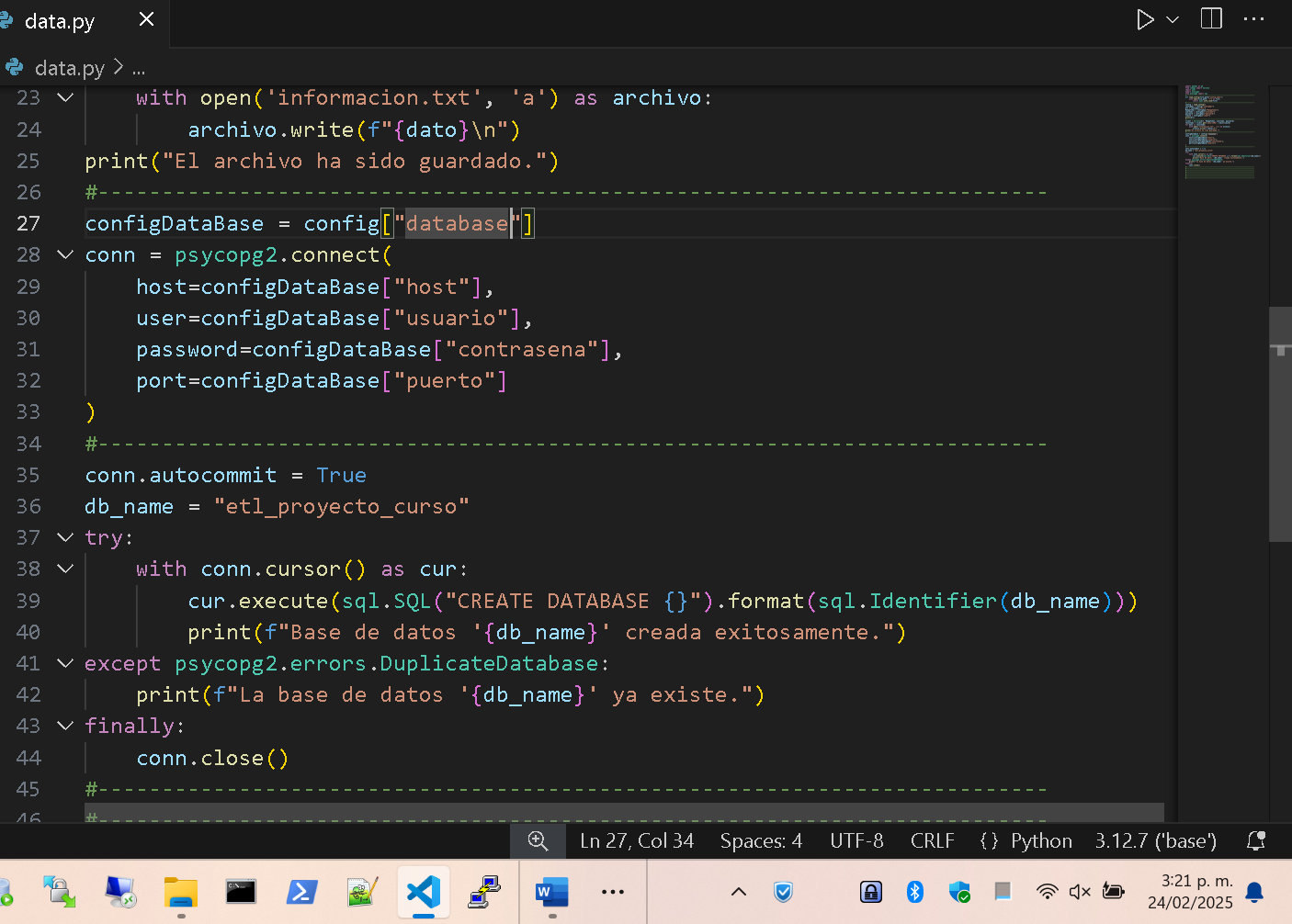
Se instala la librería psycopg2 en el entorno.



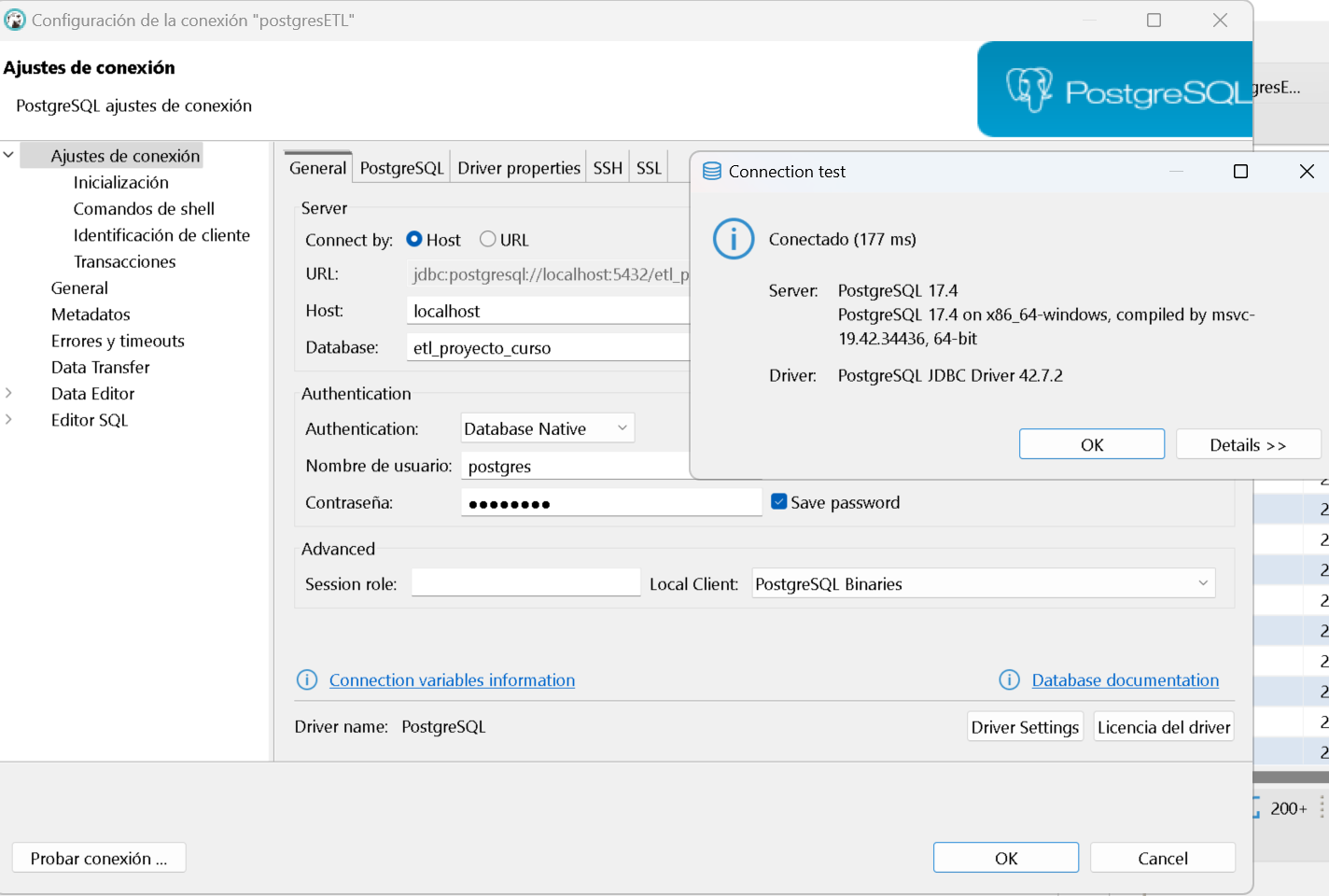
Se configura en el archivo .yaml las credenciales de conexión a la api y se configura el limite de registros de consulta.



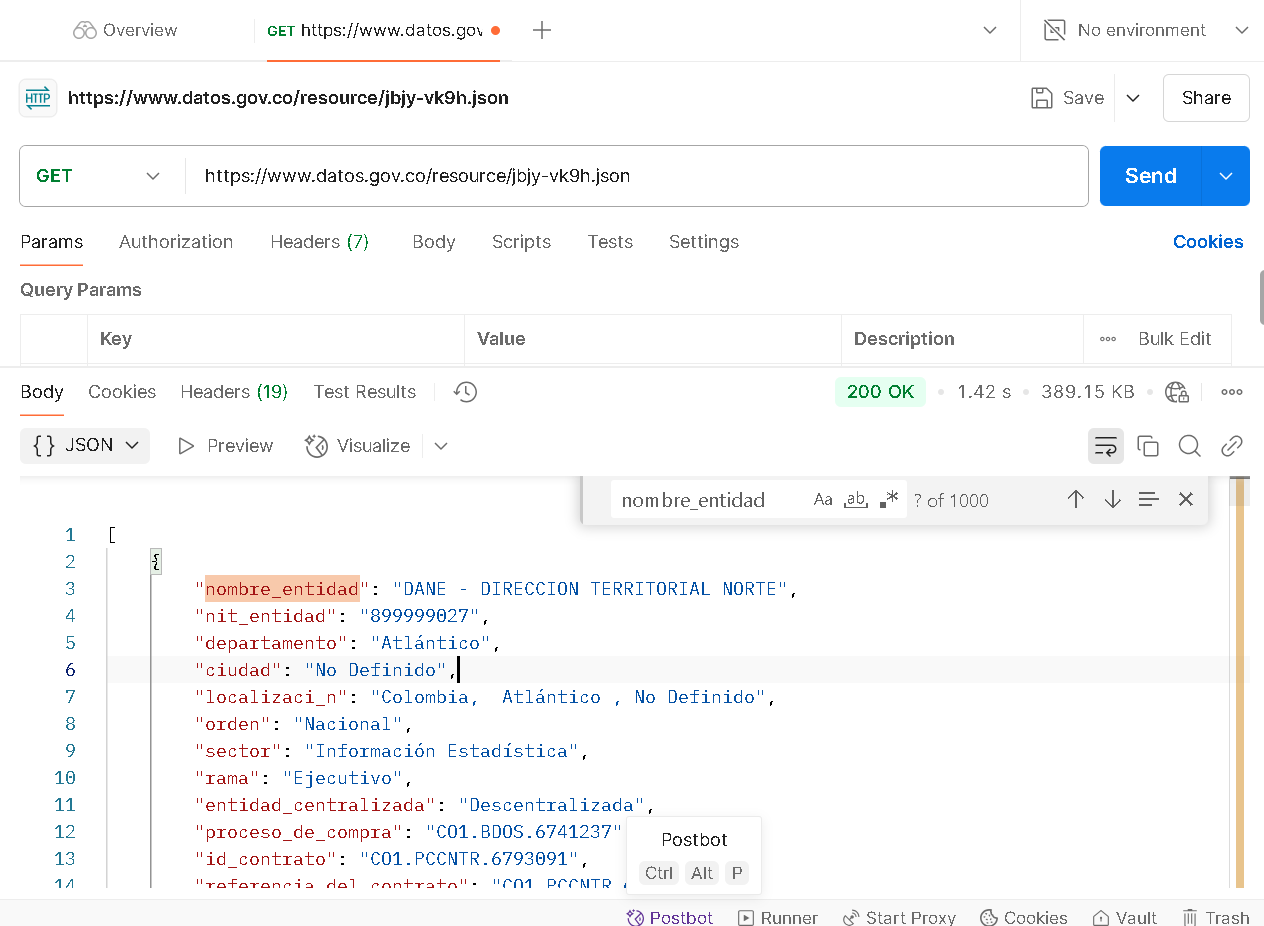
Se crea la base de datos llamada etl\_proyecto\_curso

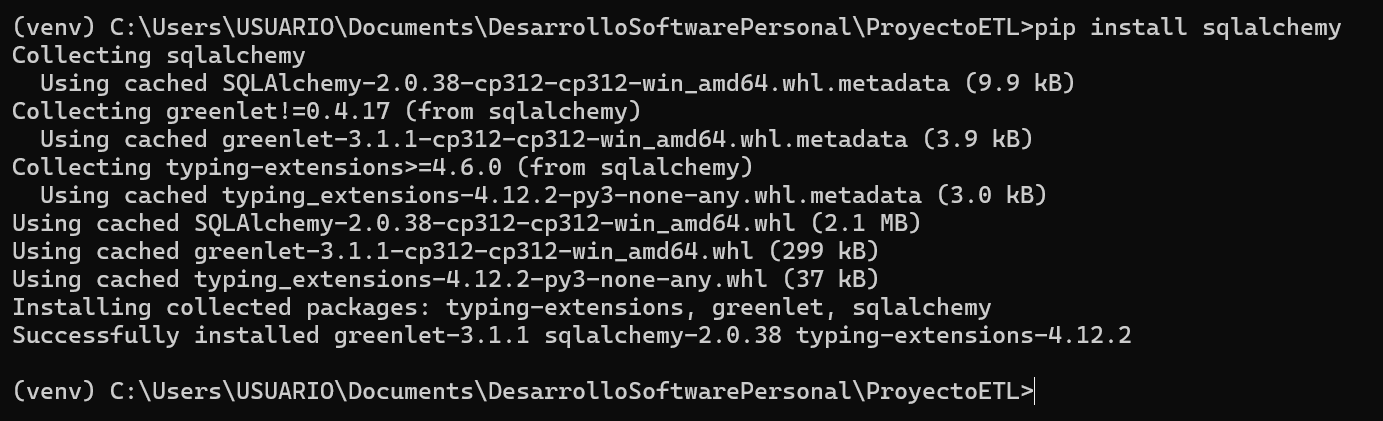


Se valida la conexión a la base de datos.

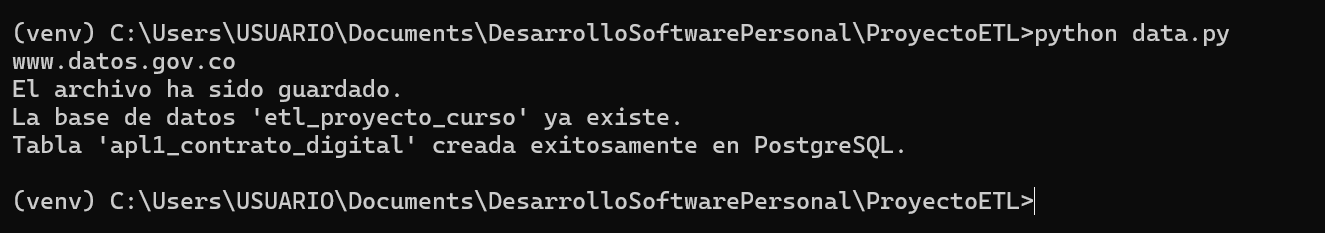


Se realiza la consulta desde postman para traer el formato json con todas las variables contenidas y definir las que serán usadas con nombre completo en el momento de crear la tabla.

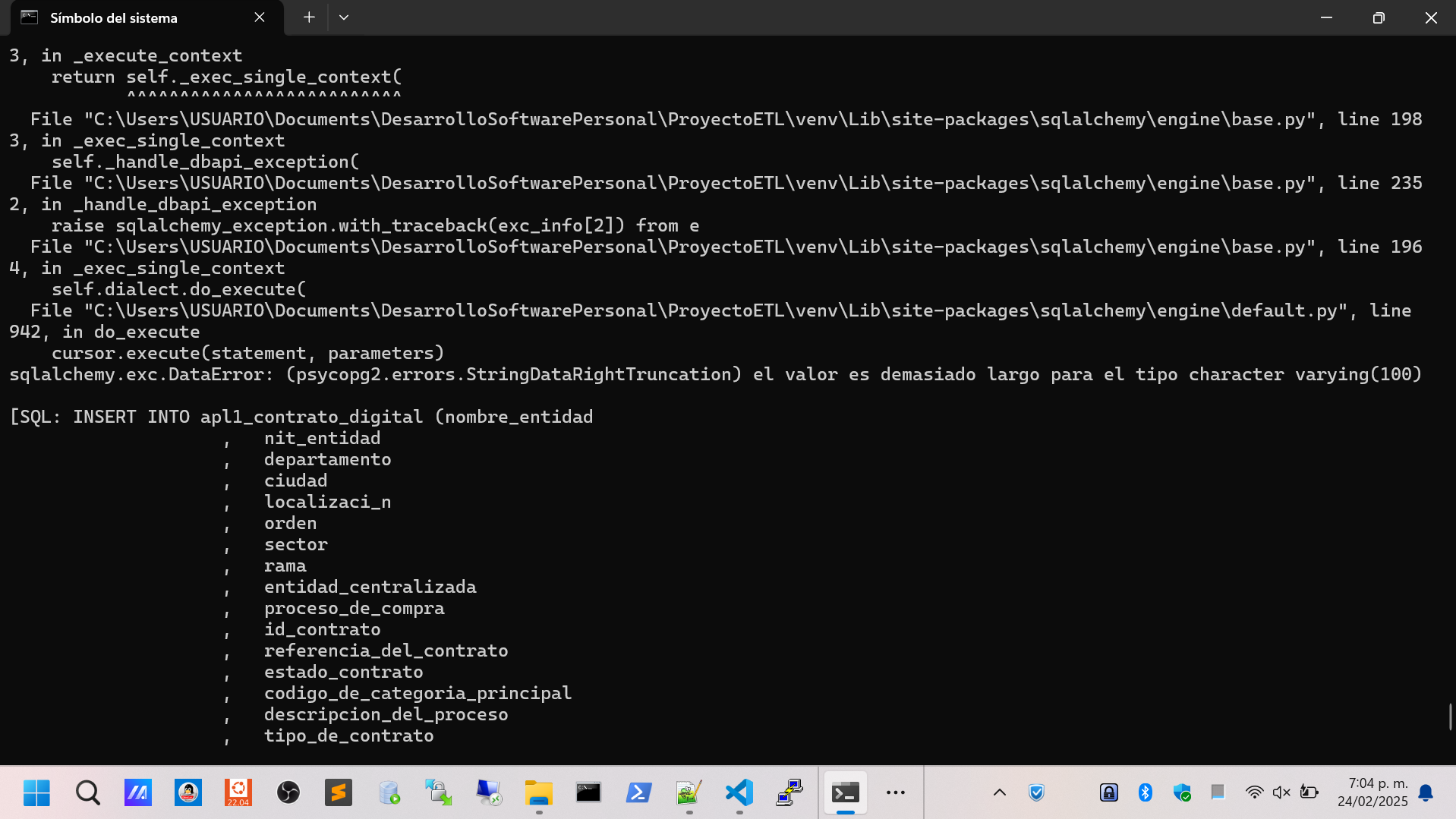


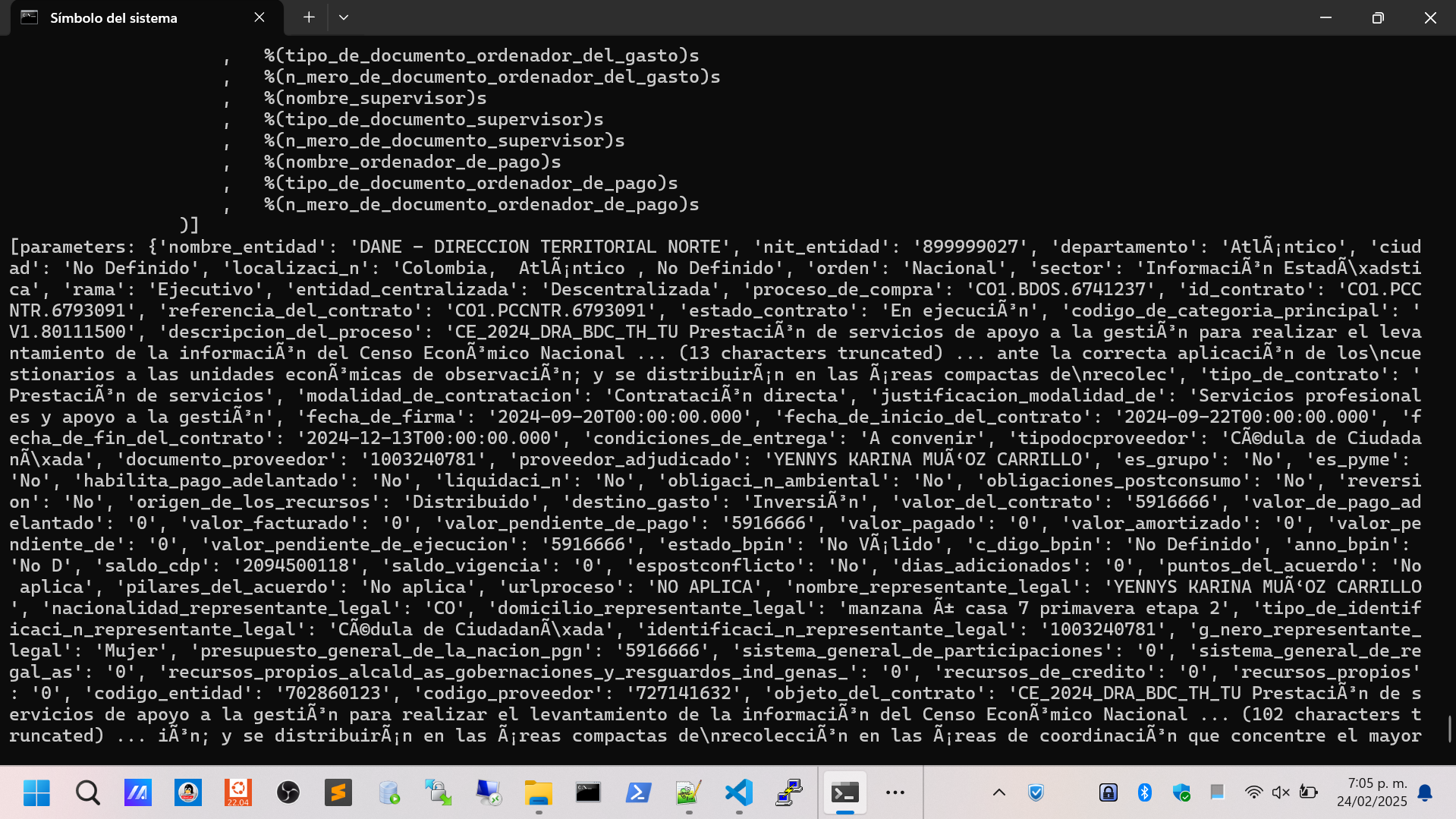


Se crea la tabla apl1\_contrato\_digital correctamente.

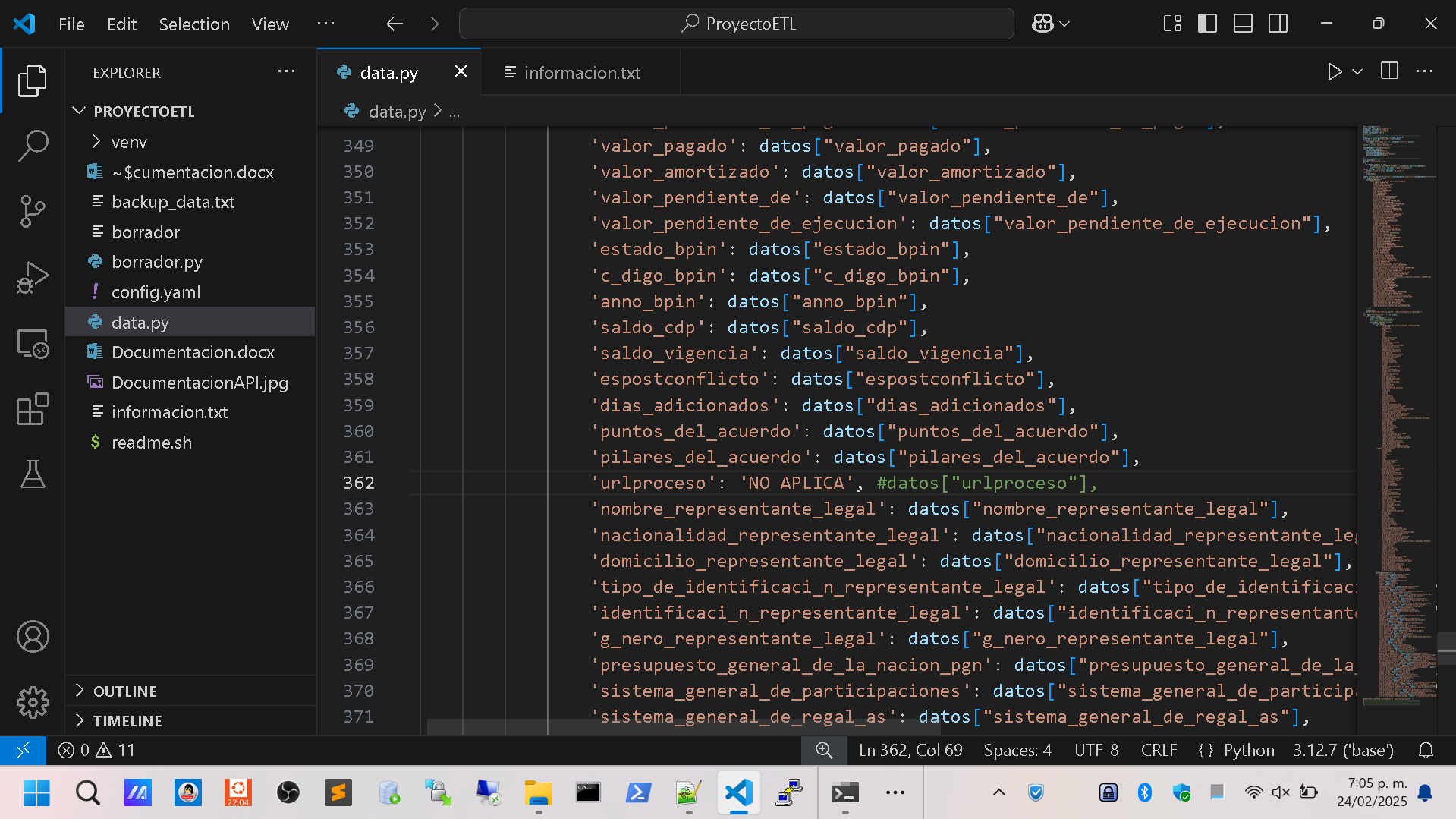


Se crea el query para realizar los insert y se relacionan los datos del archivo de configuración, empezamos a visualizar los posibles errores que se pueden presentar.

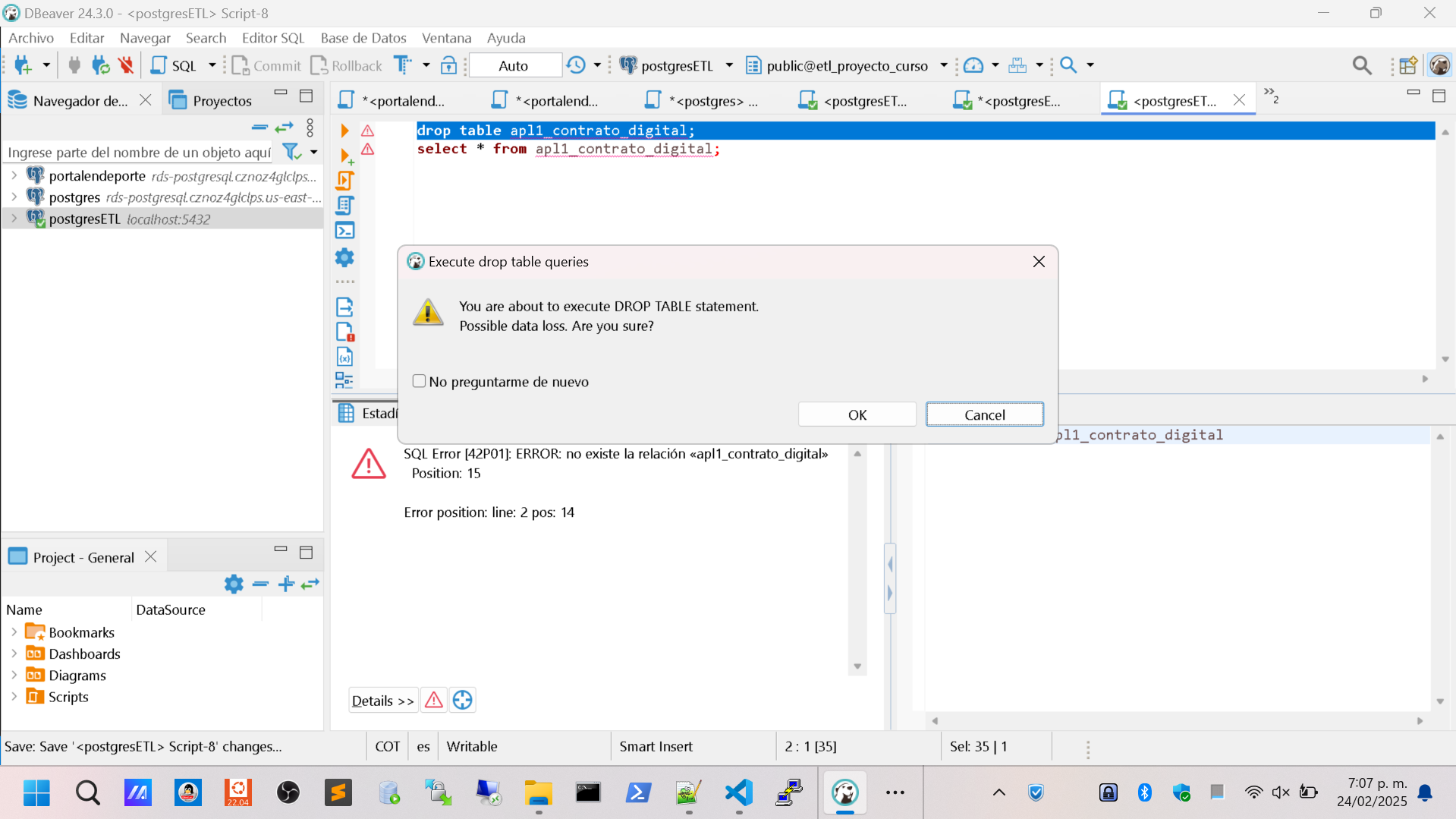




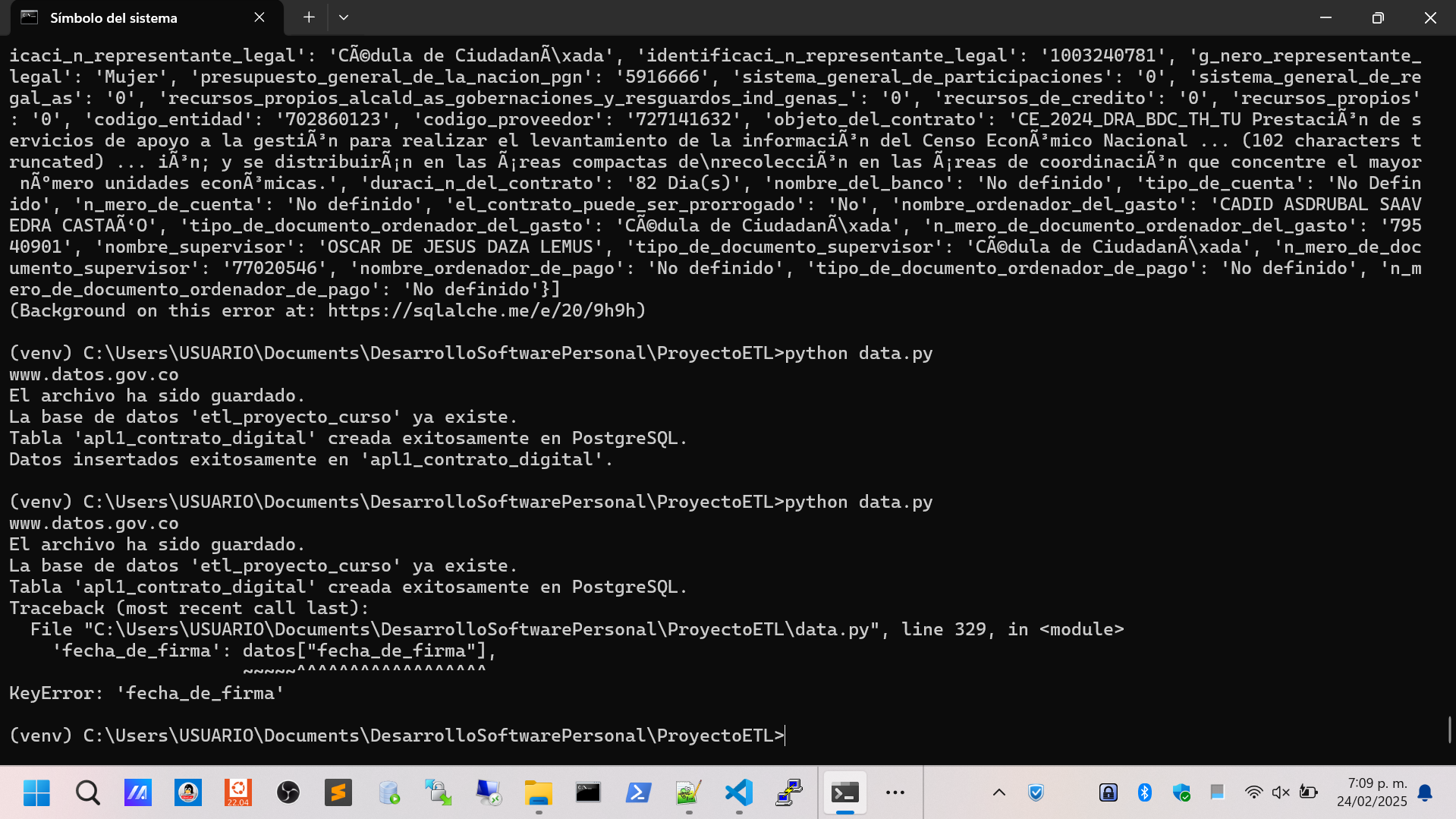
Se ignoran algunos campos debido al tamaño del texto por cada variable.



Se borra la tabla para ampliar los campos nuevamente.

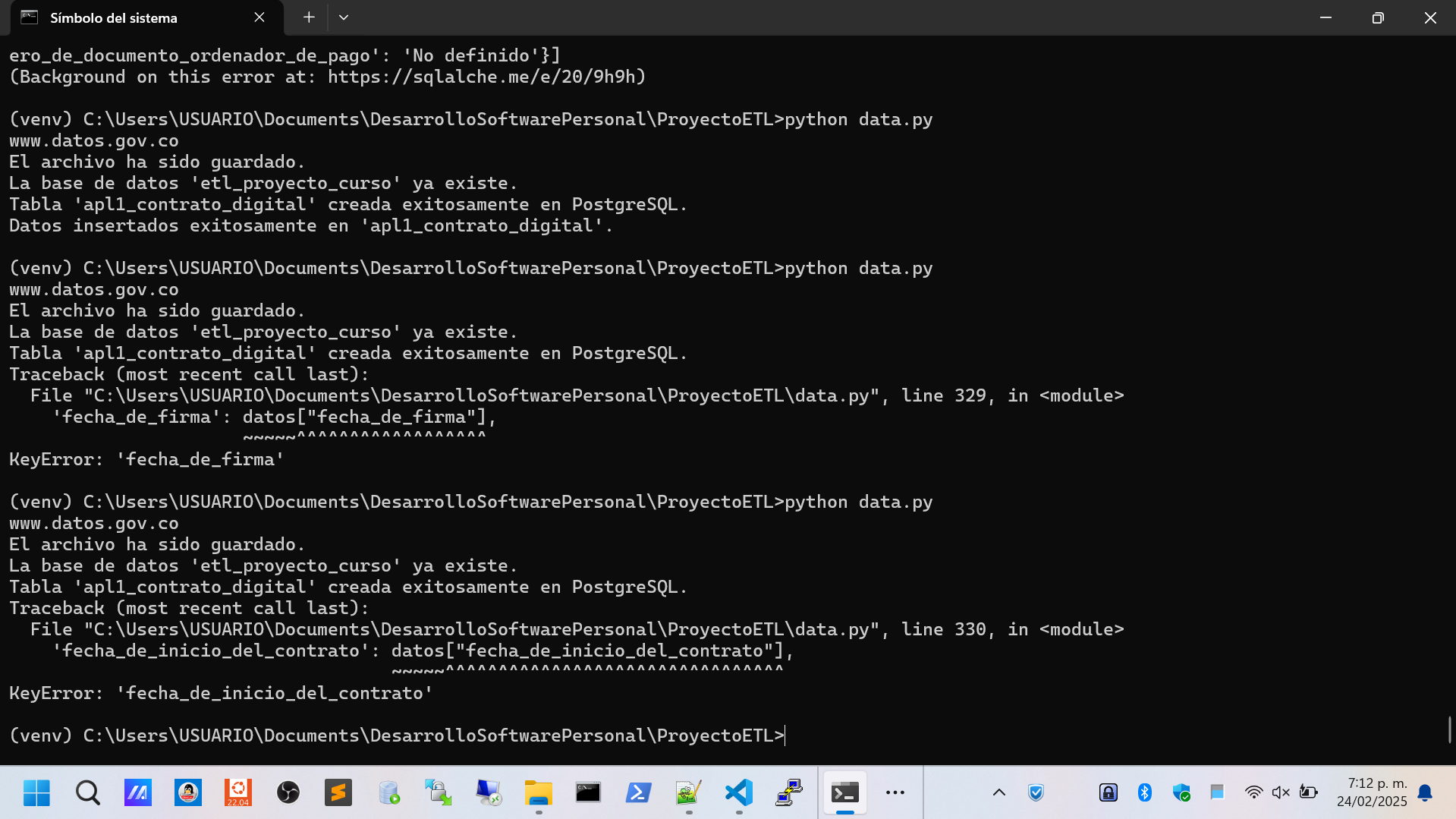


Se crea nuevamente la tabla ampliando los campos de tamaño y se procede a realizar el insert de los registros.

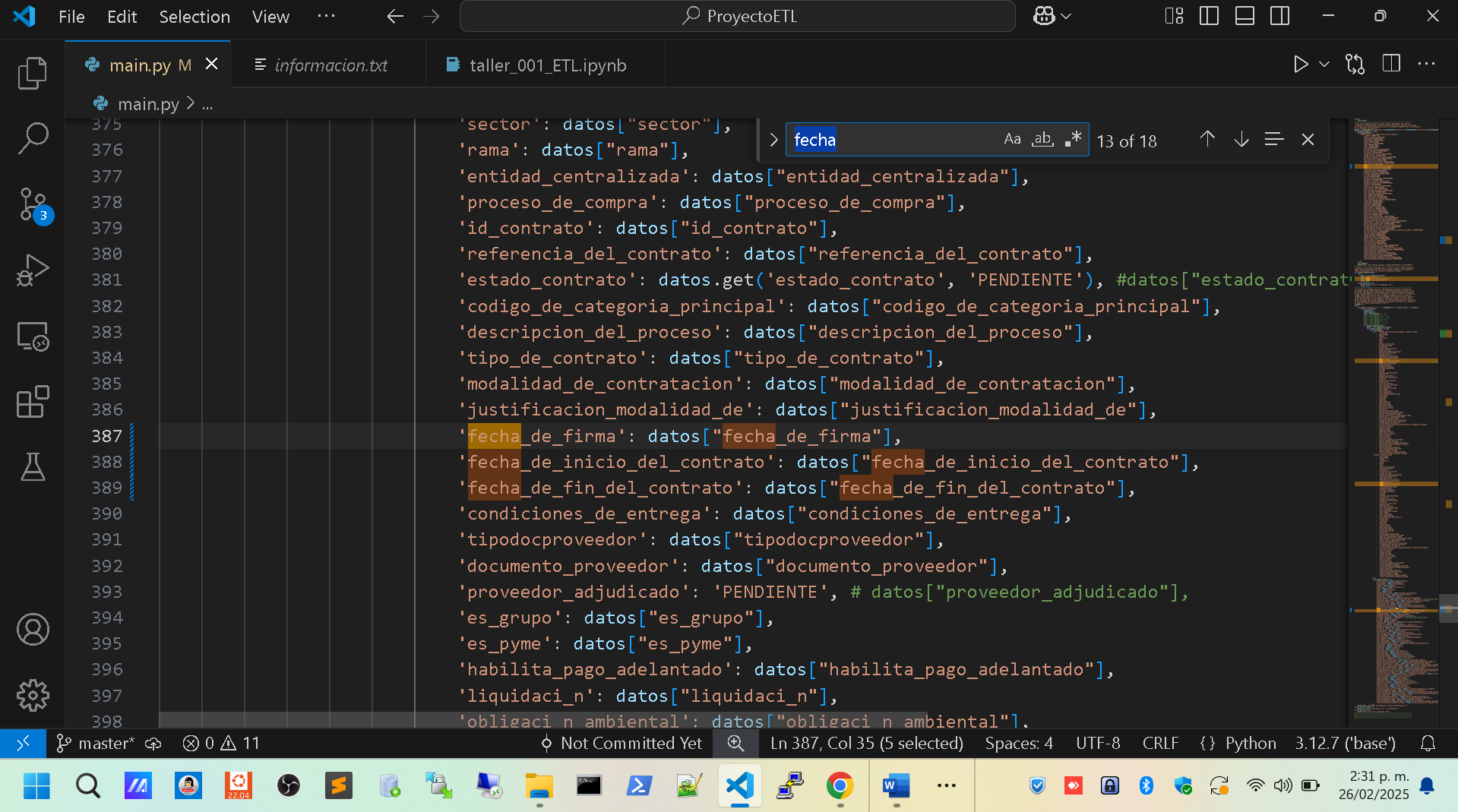


Se procede a revisar algunos inconvenientes con las fechas y se procede a corregir los errores.

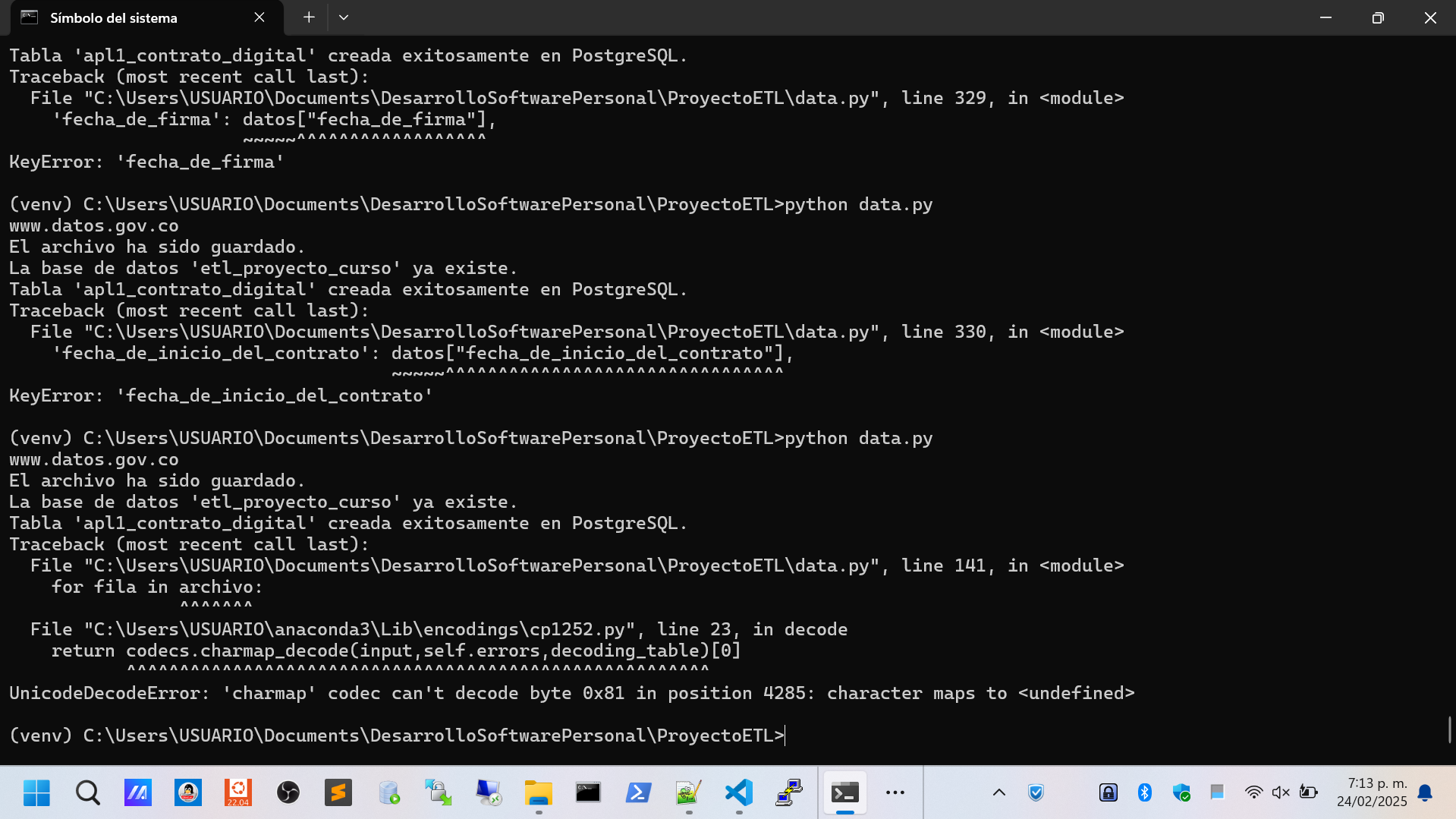


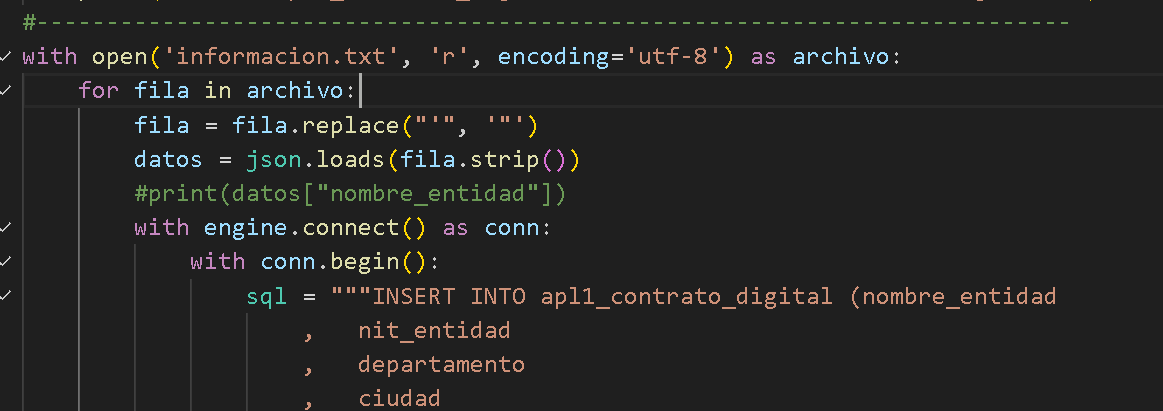


Se dejan pendientes por el momento mientras se define el formato de fechas.



Surgen otros errores al mapear los formatos json y se procede a revisar las causas para ser corregido.





Traceback (most recent call last):

File "C:\Users\USUARIO\Documents\DesarrolloSoftwarePersonal\ProyectoETL\data.py", line 141, in <module>

for fila in archivo:

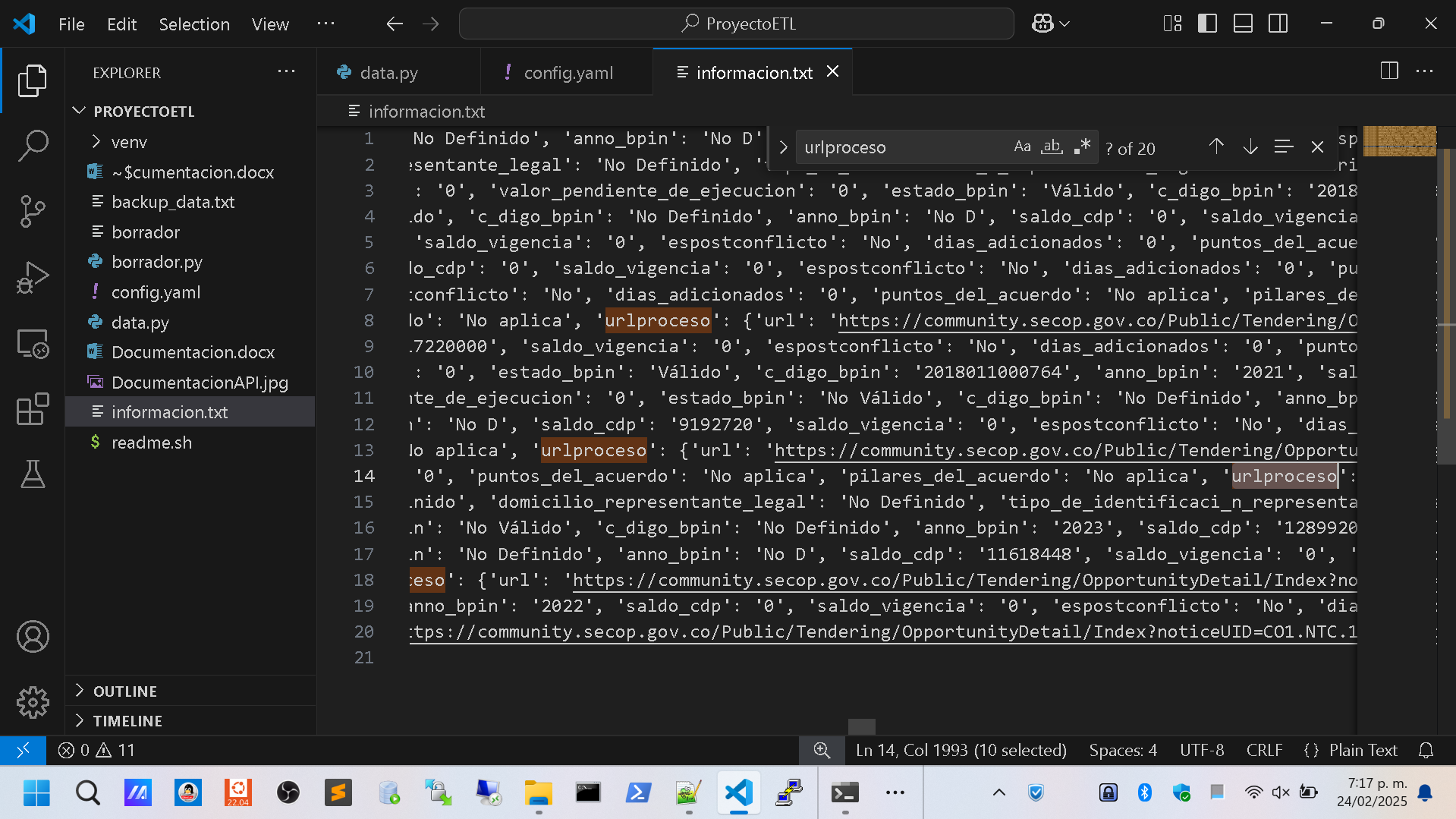
^^^^^^^

File "C:\Users\USUARIO\anaconda3\Lib\encodings\cp1252.py", line 23, in decode

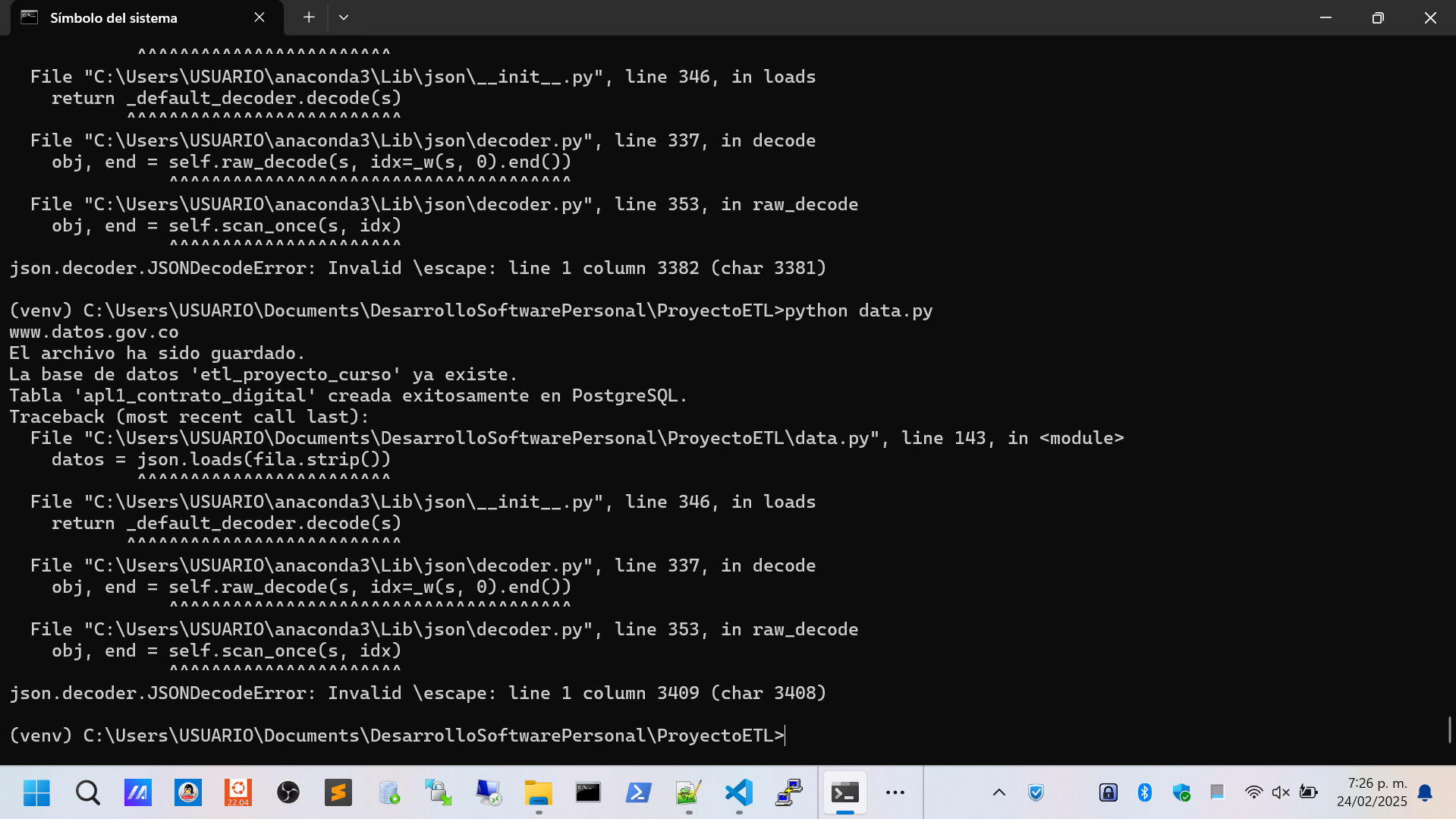
return codecs.charmap\_decode(input,self.errors,decoding\_table)[0]

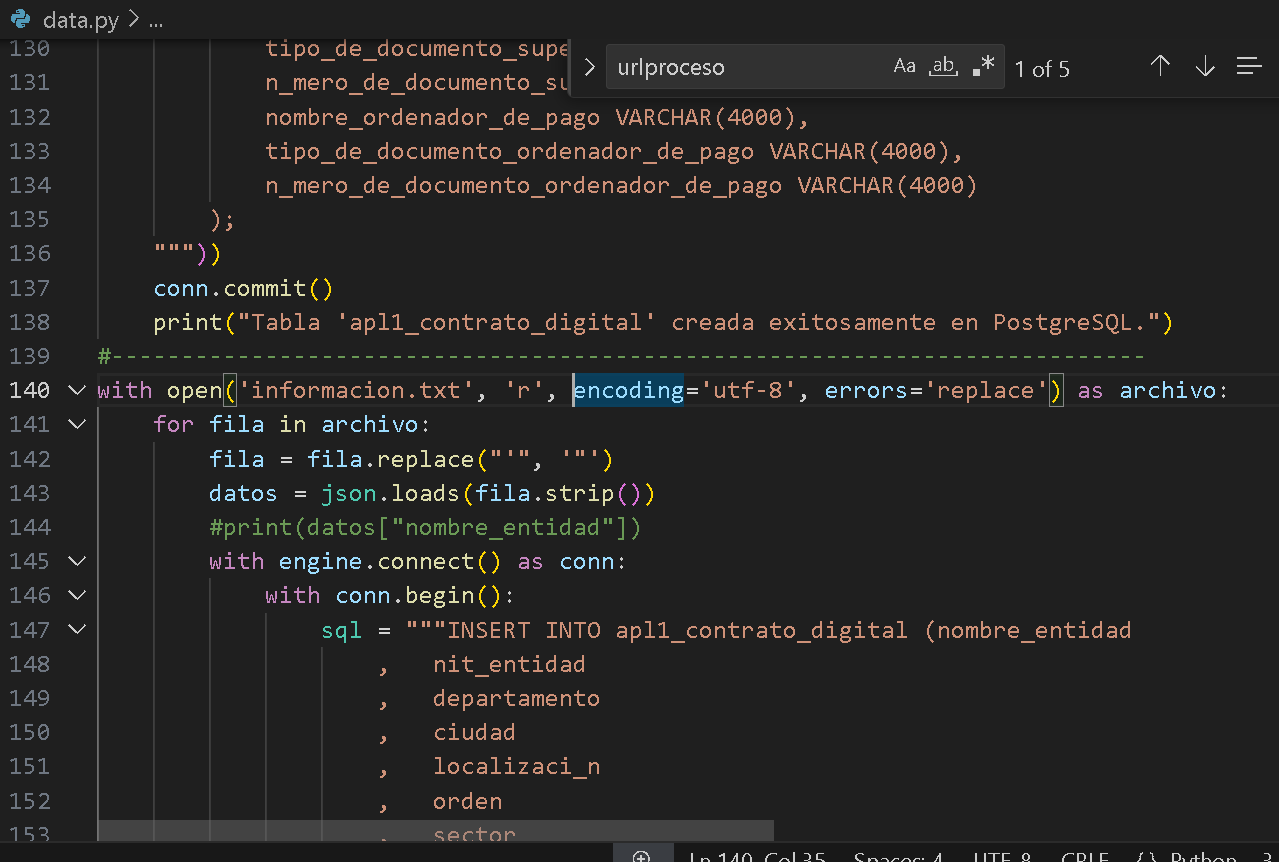
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

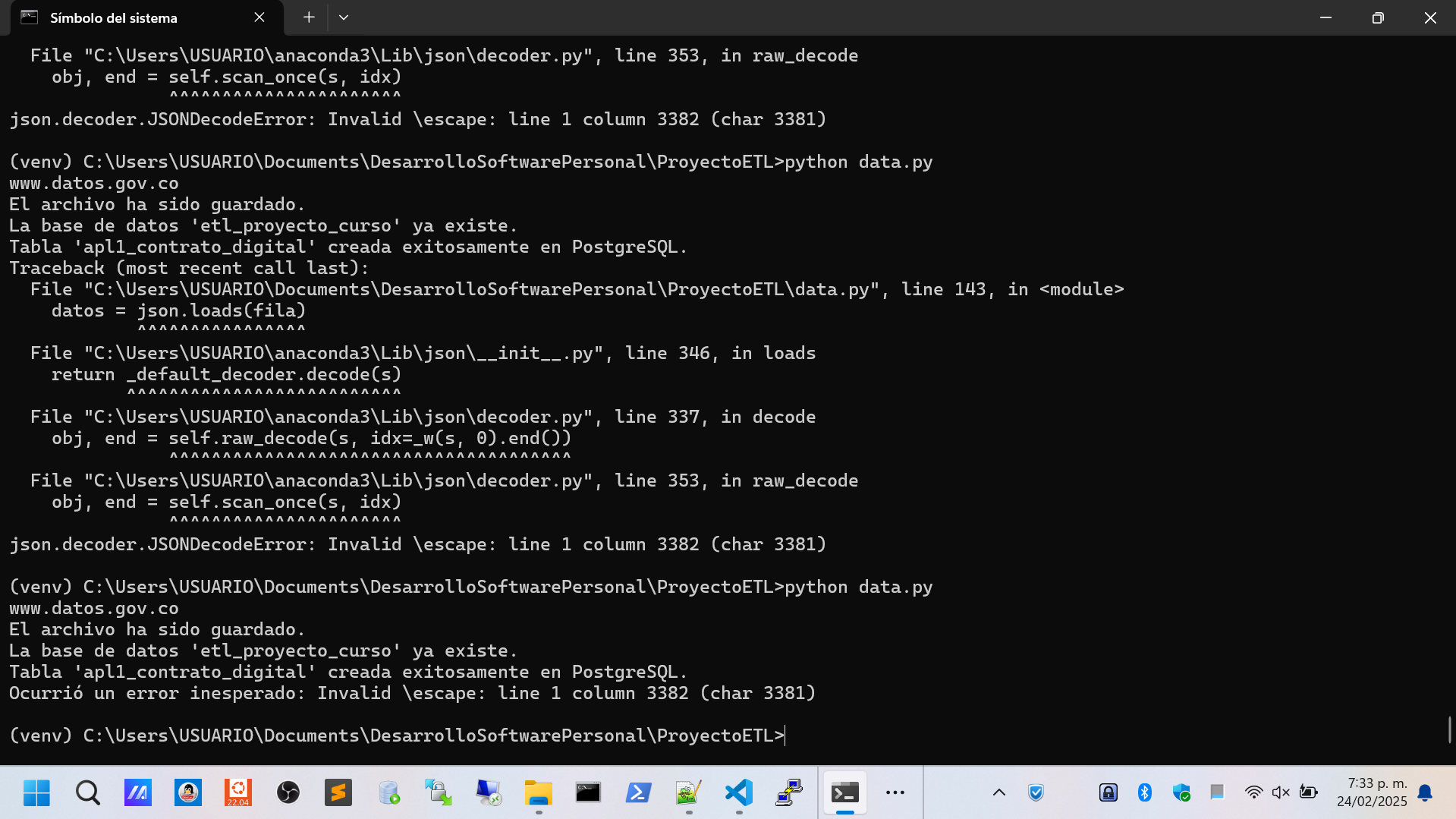
UnicodeDecodeError: 'charmap' codec can't decode byte 0x81 in position 4369: character maps to <undefined>

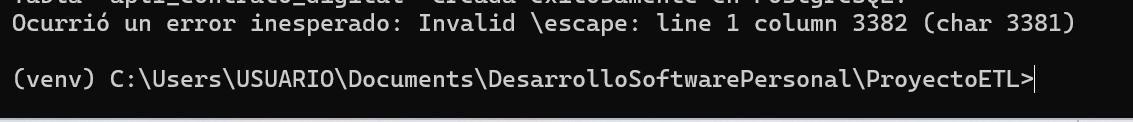


Se encuentra que en algunos casos dentro de una llave viene un objeto como valor, se procede a dejar pendientes estos valores por el momento.

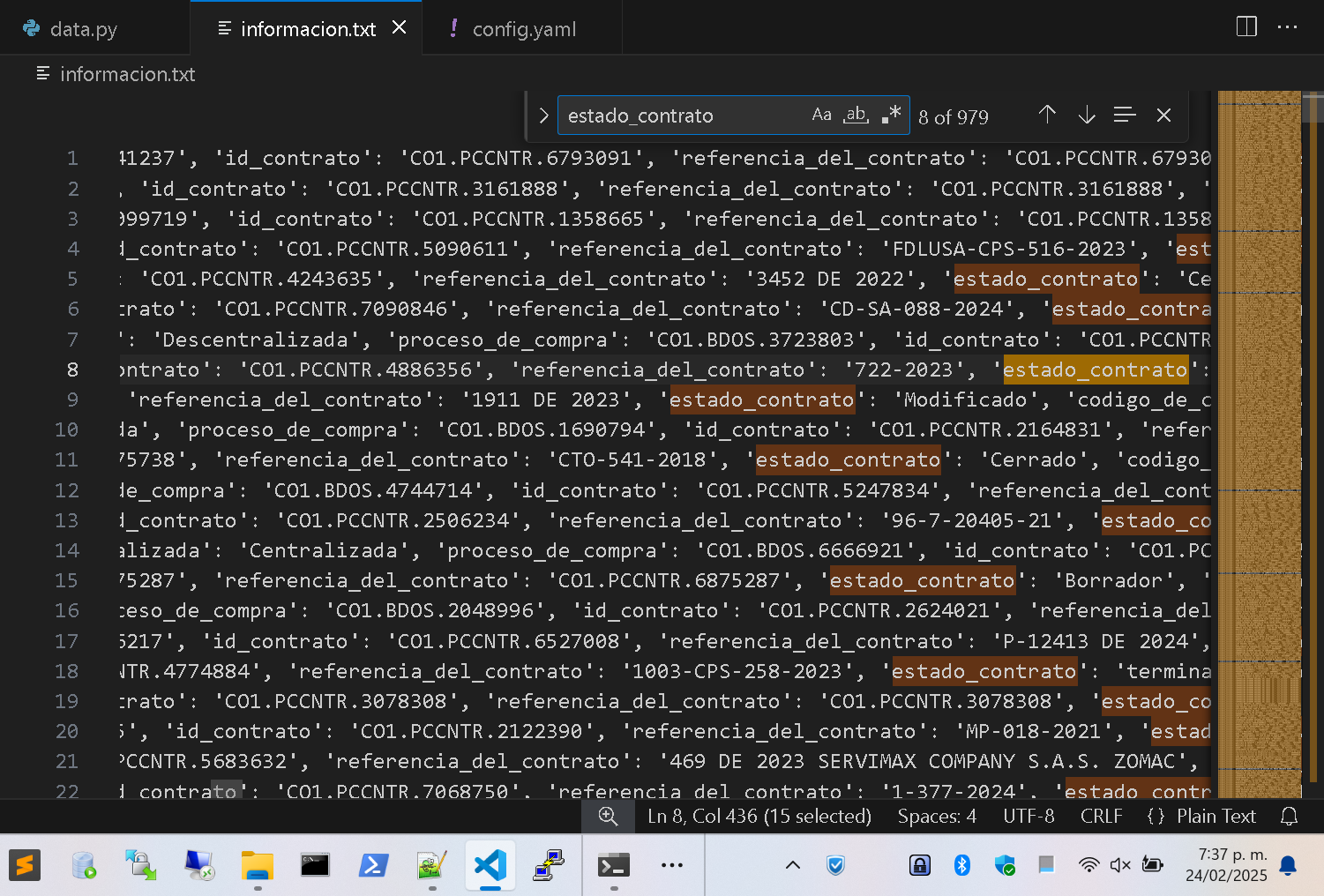




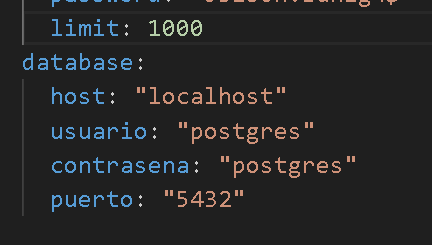


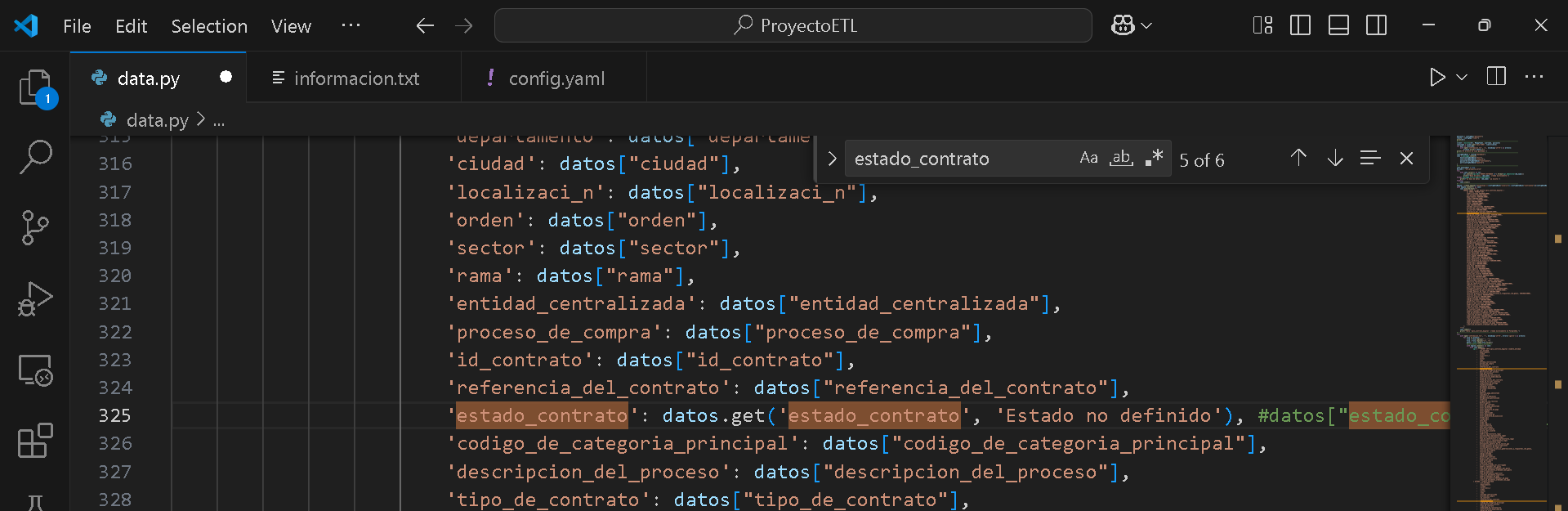


Se encuentra que en algunos casos las llaves no coinciden con el número de registros, es decir vienen faltantes.

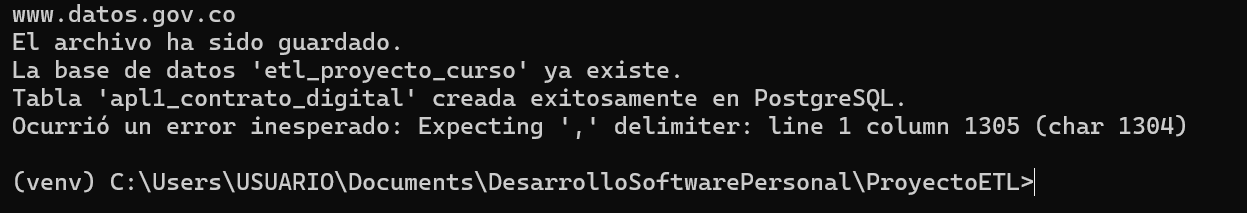


Se define un limite de carga para realiza pruebas del proceso de extracción, por el momento un limite de 1000 registros para continuar con el proceso de pruebas de extracción.

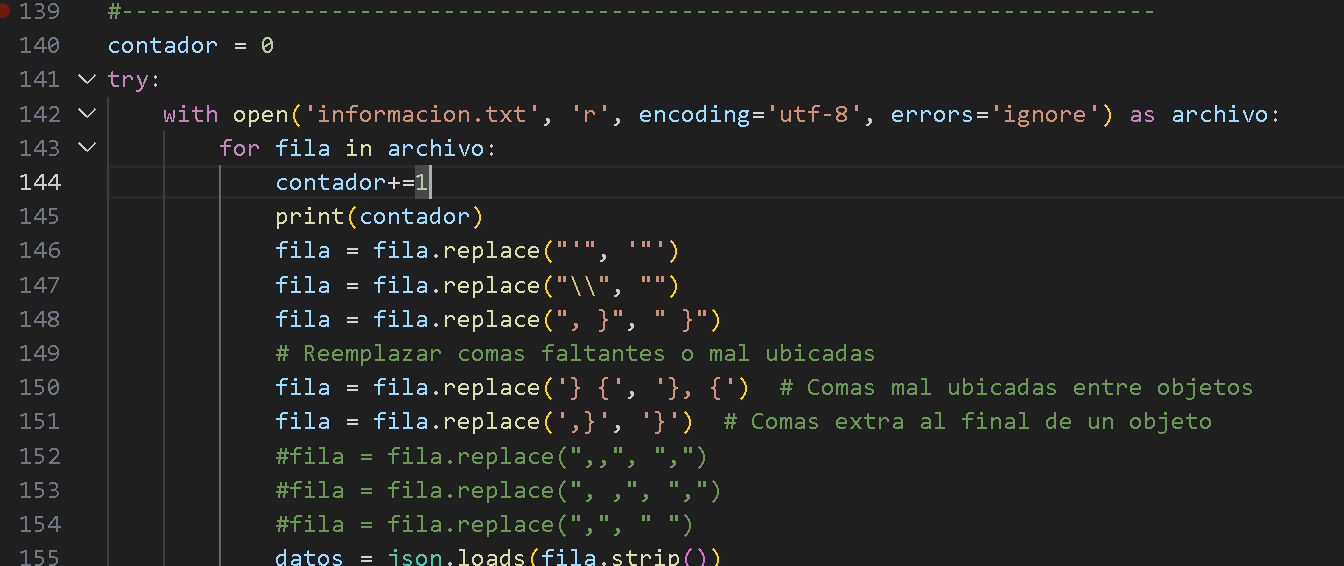


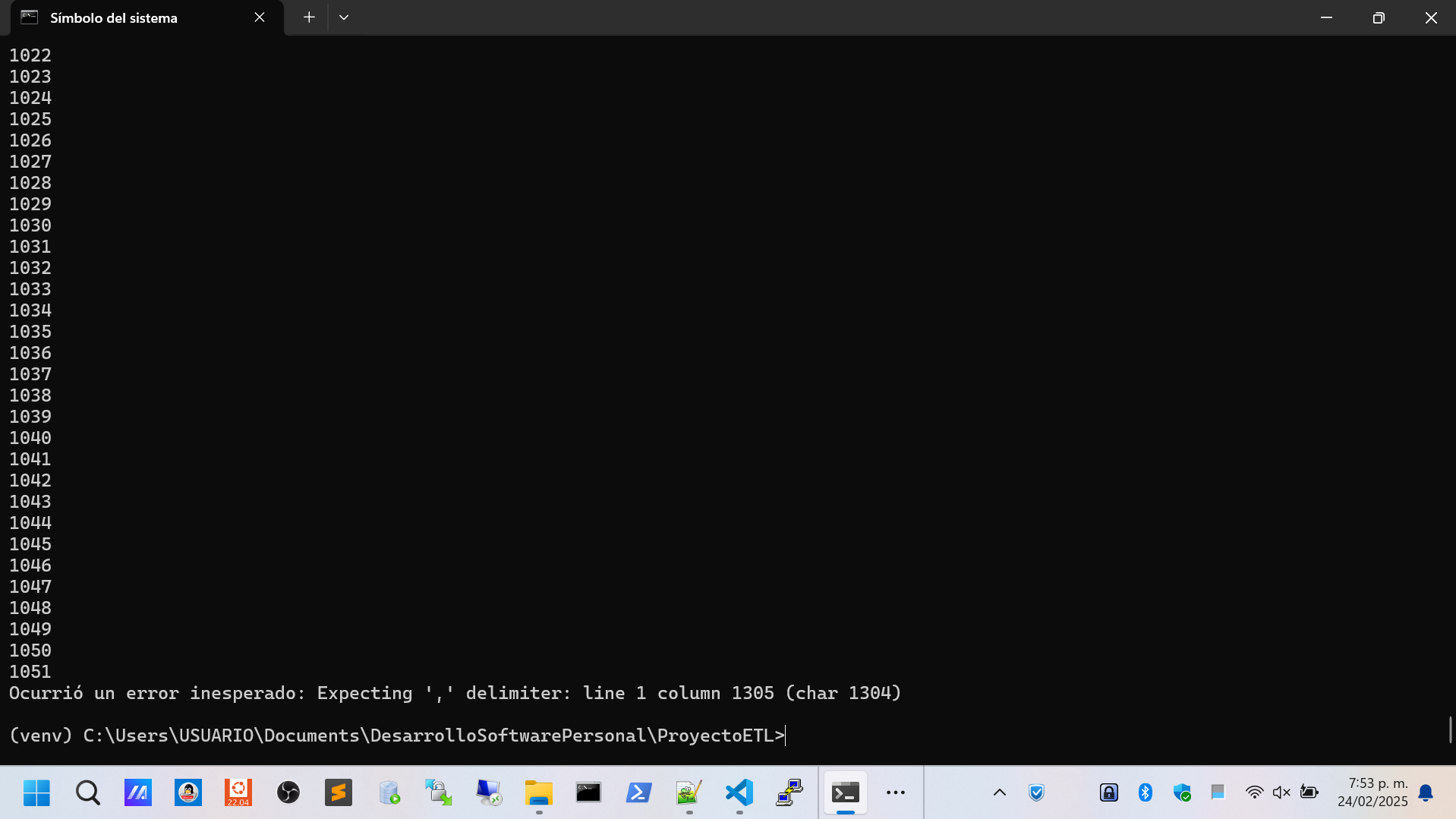


Encontramos errores con las comillas simples y se procede a revisar los casos.



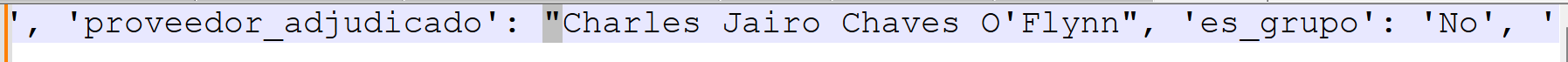
Se empieza revisando cada objeto json y se procede a corregir los errores.

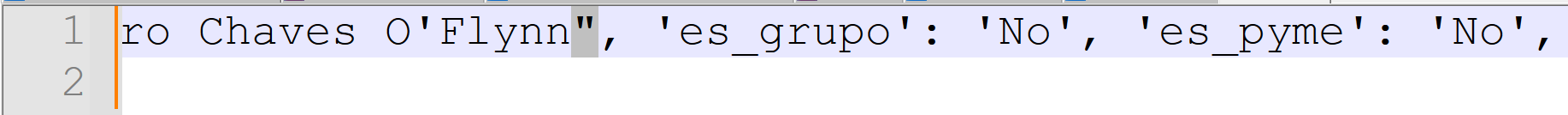


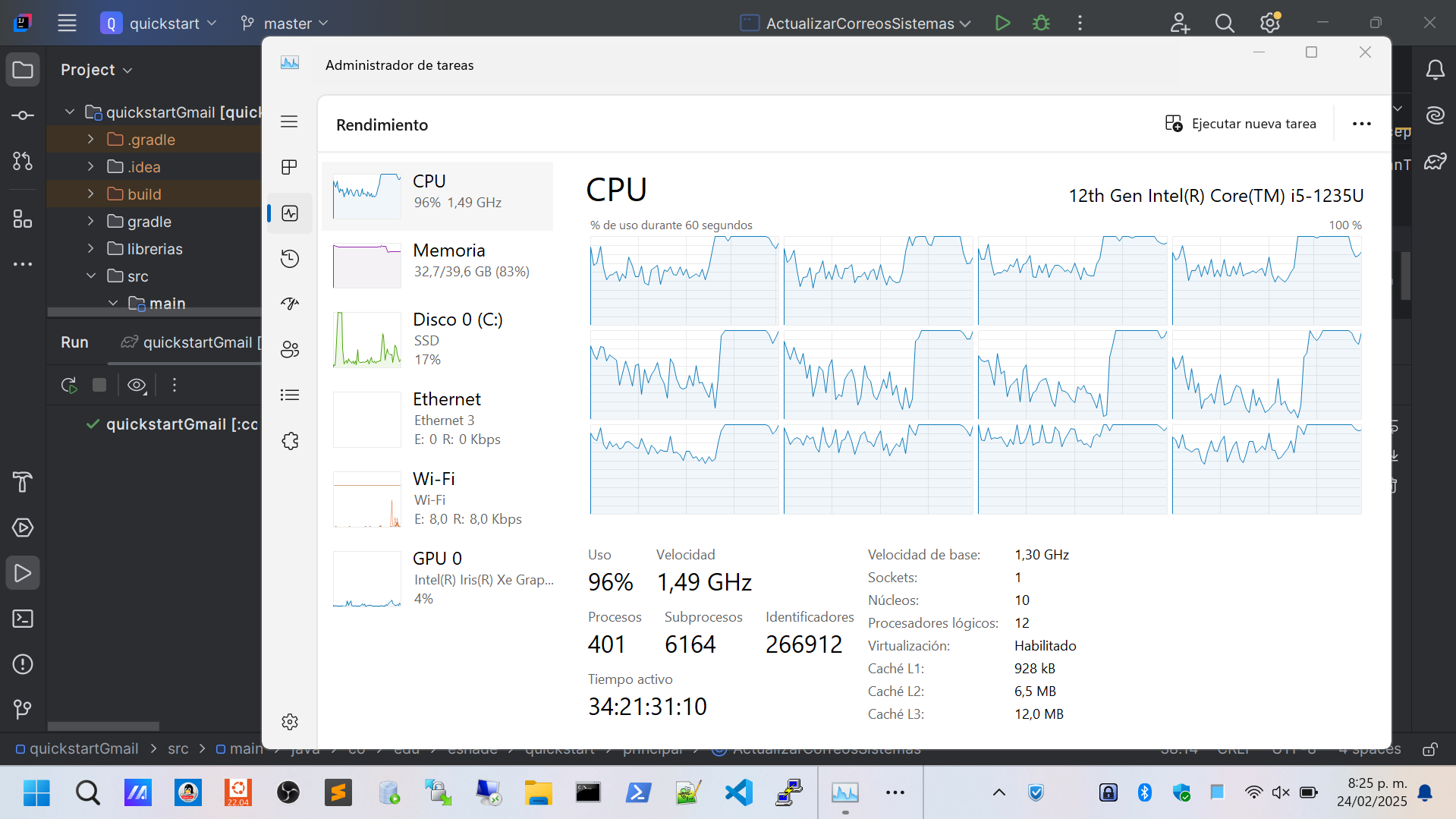


Empieza el proceso de inserción pero falla por las comillas simples.

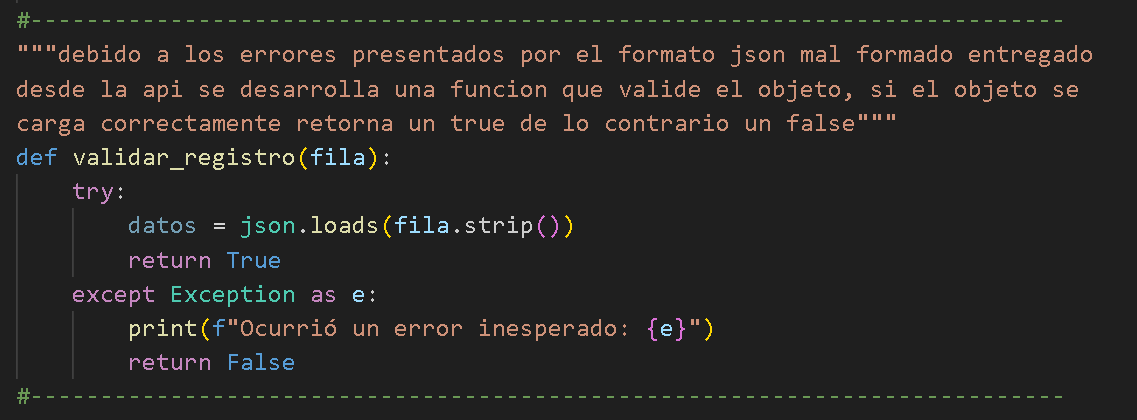


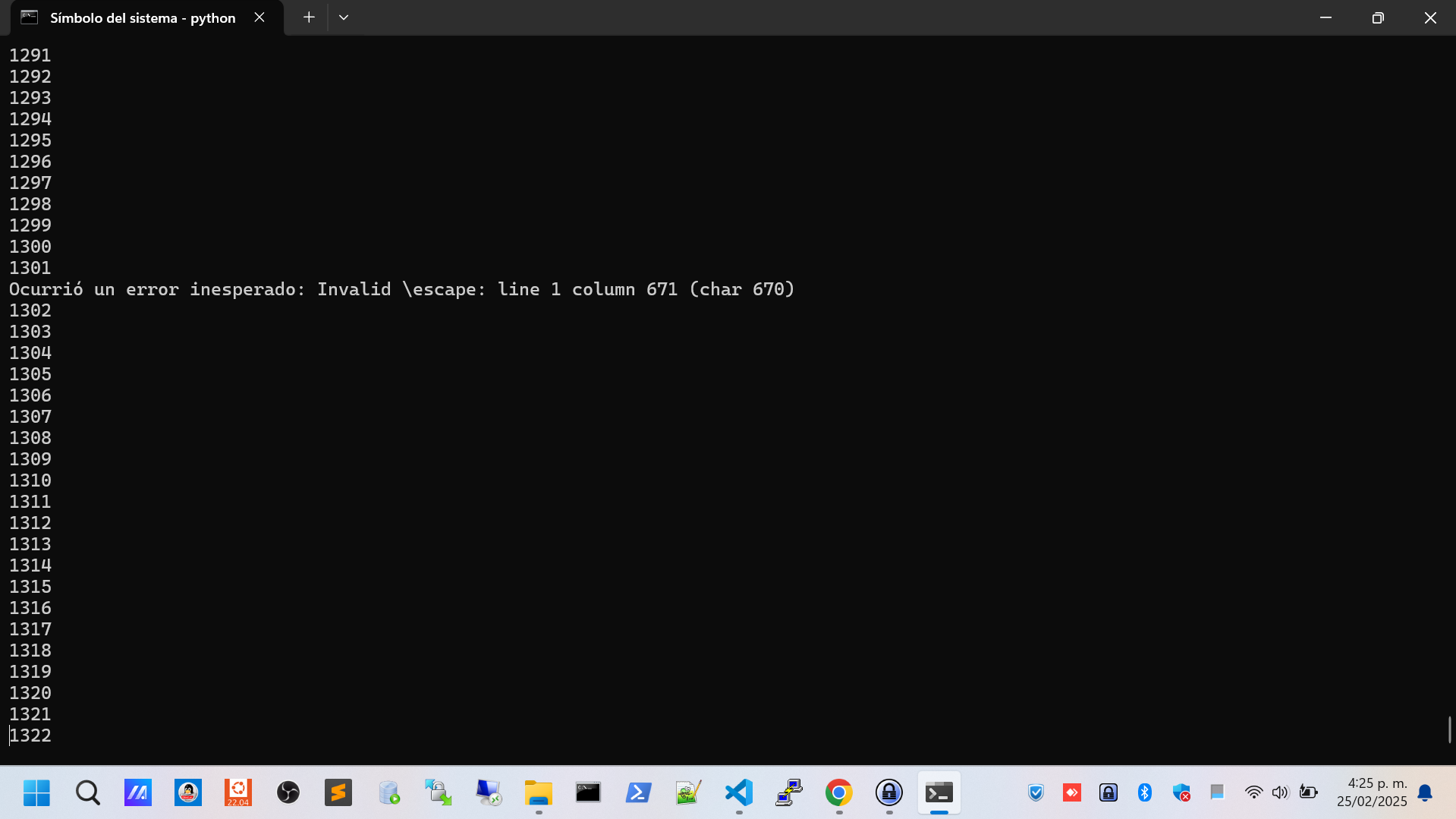


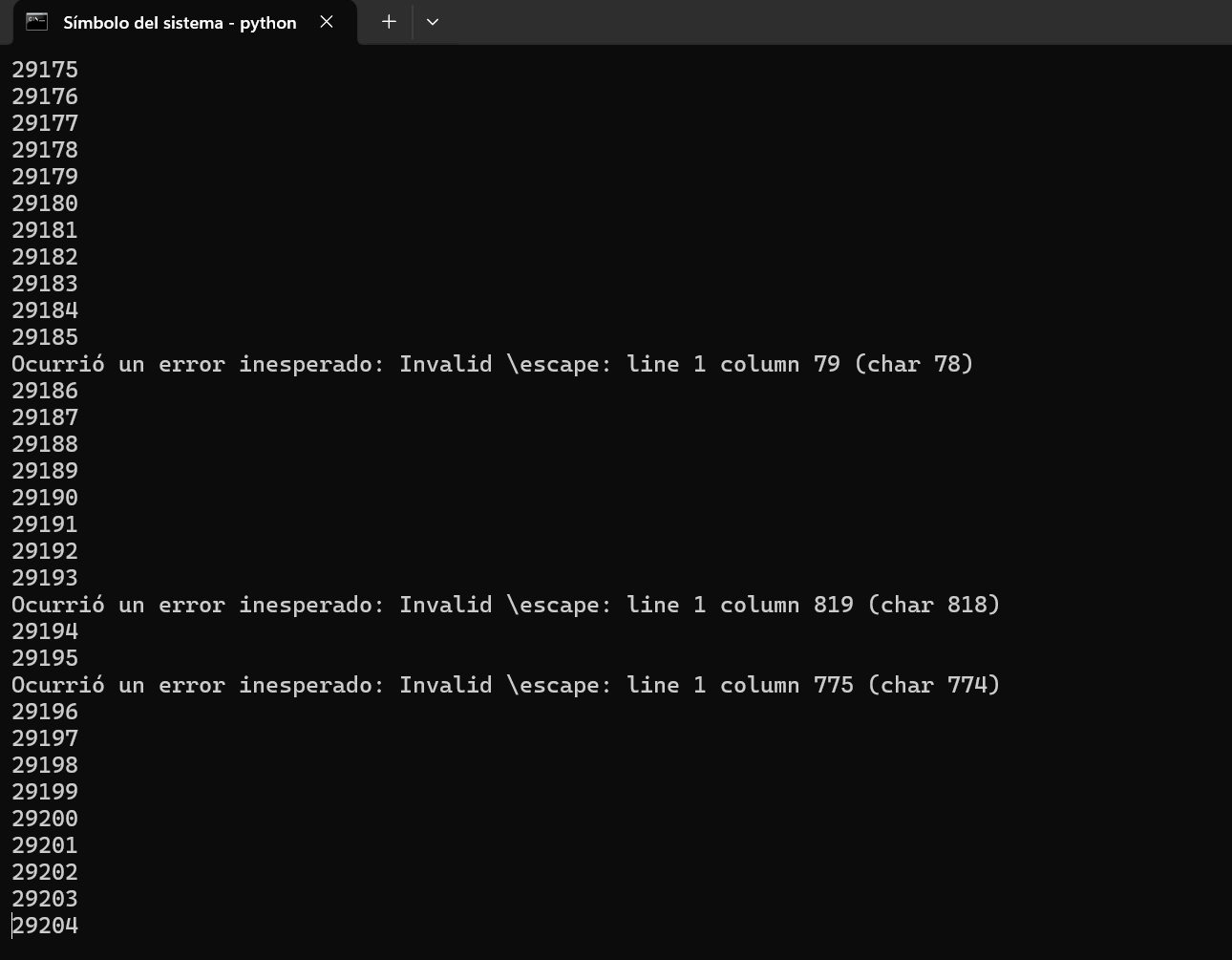


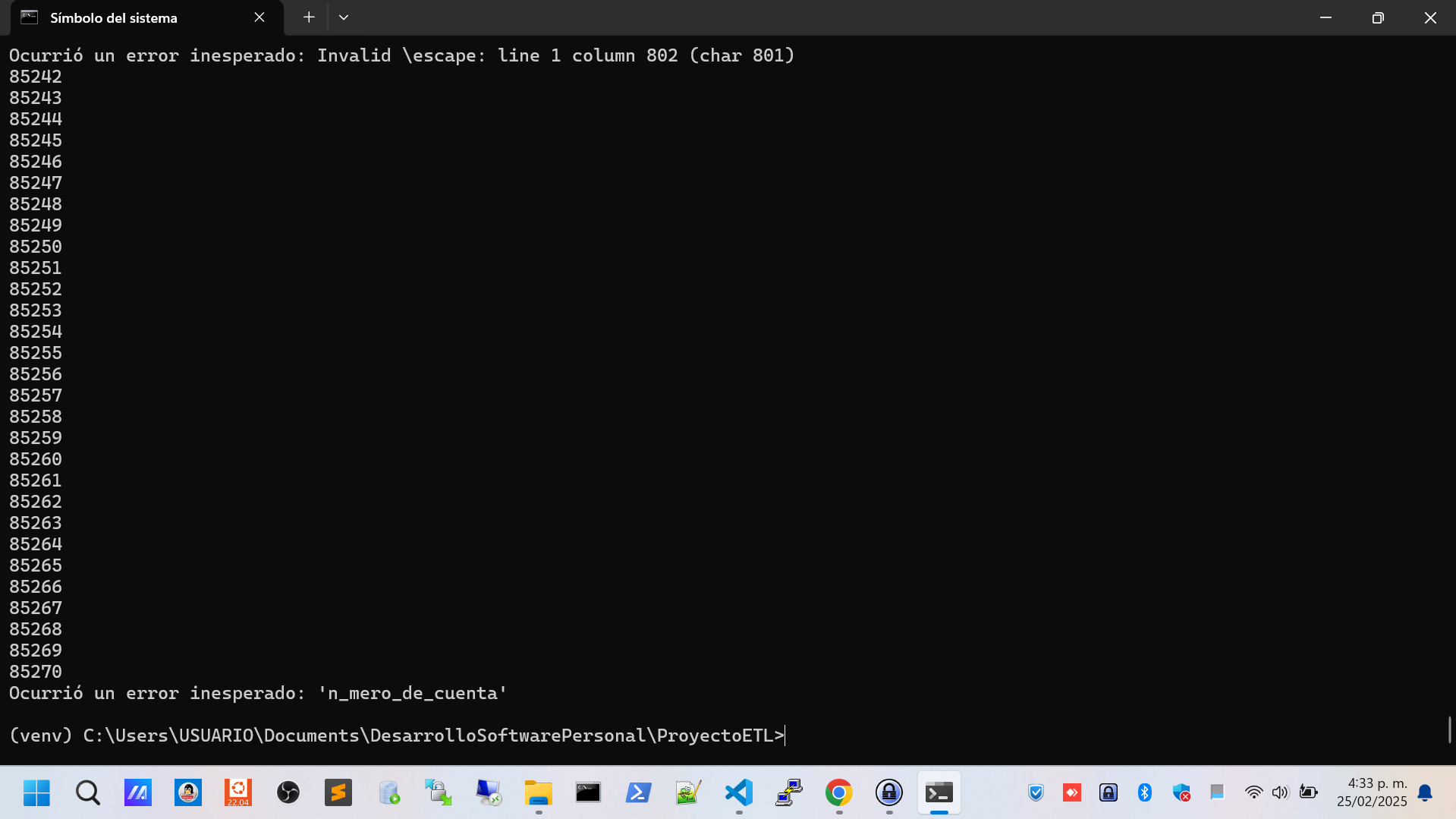


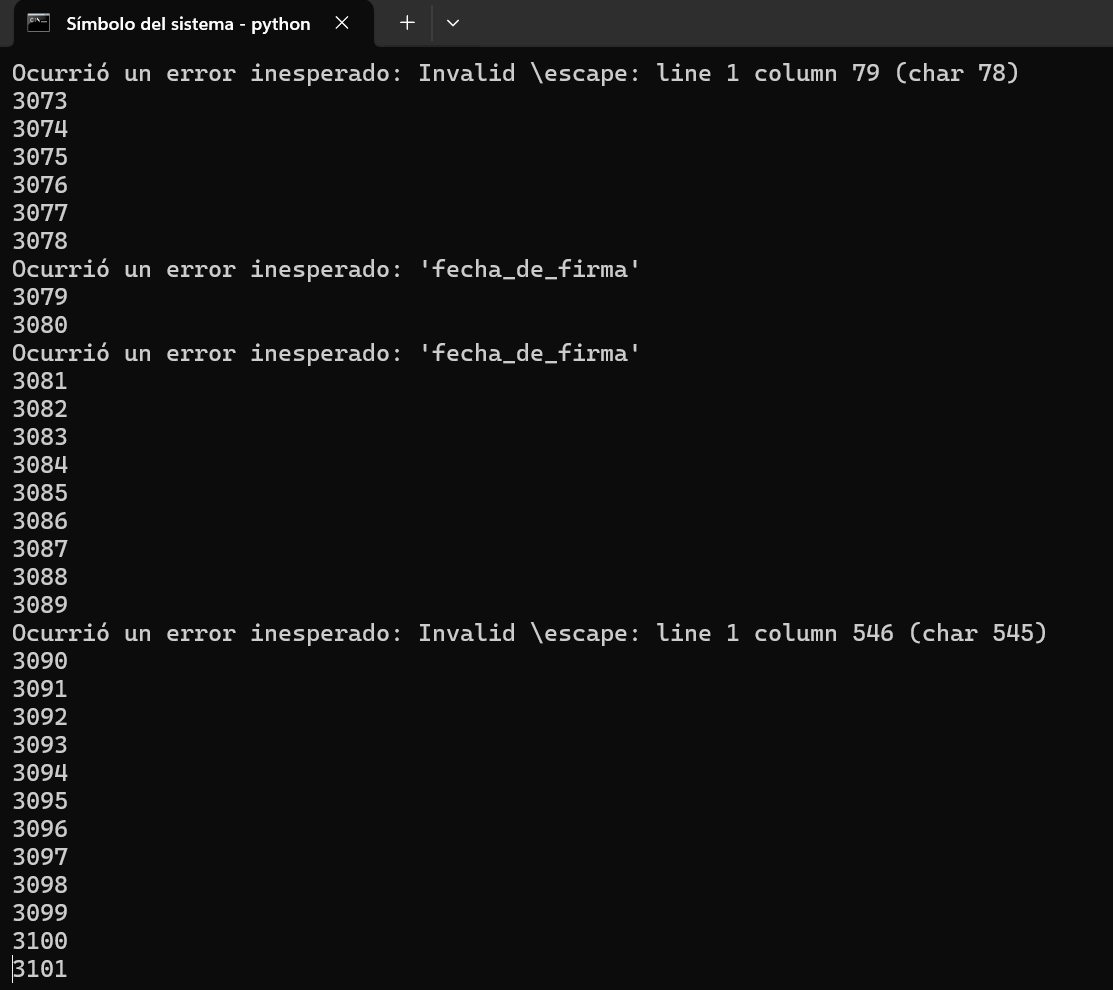
Se crea un método para validar que los registros estén correctamente en cada ciclo de lectura y se evita la detección del proceso para continuar con la inserción en la base de datos.

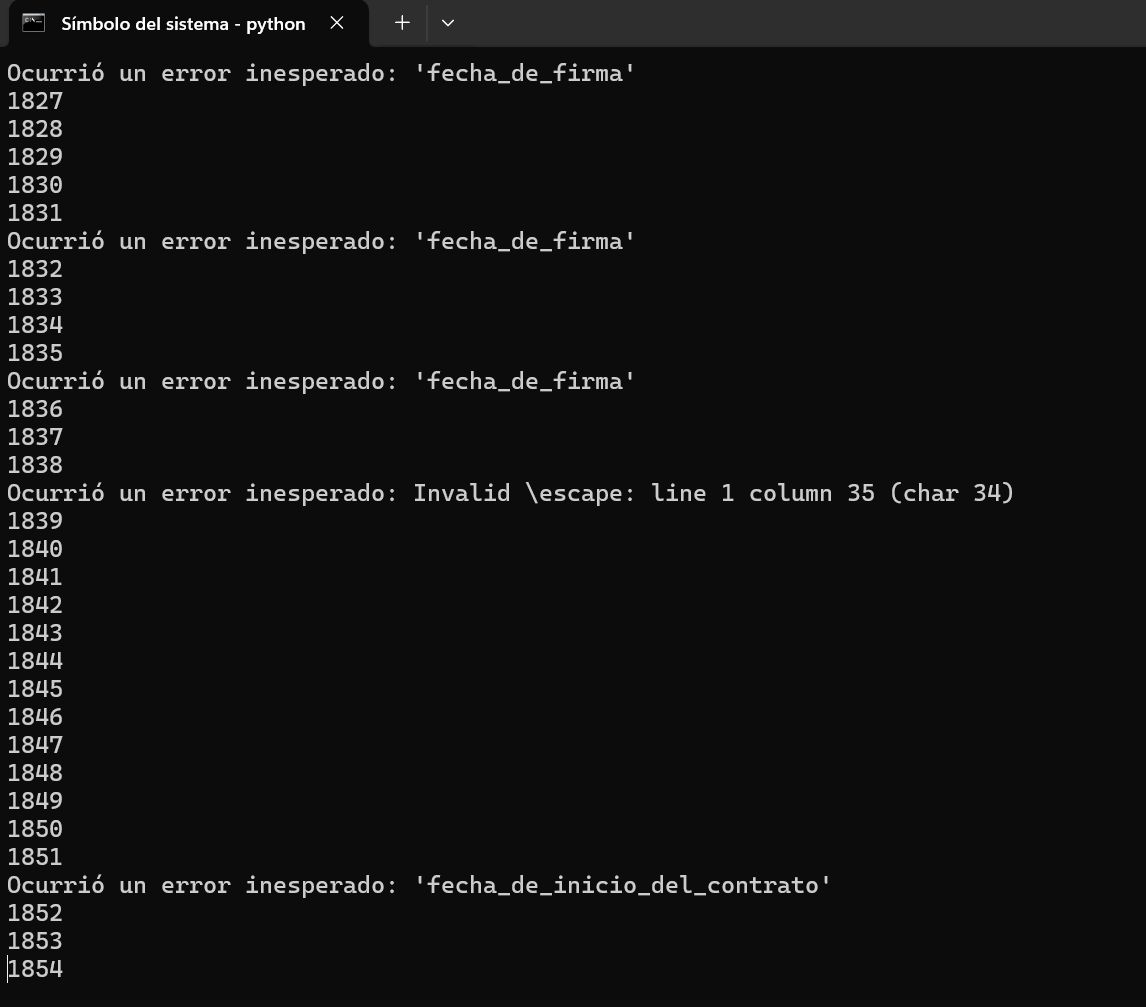




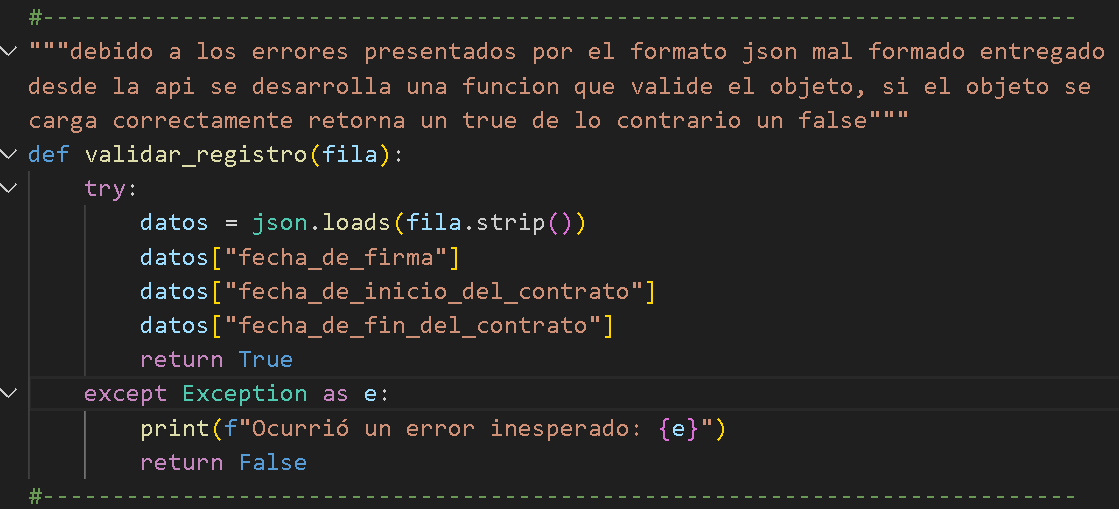




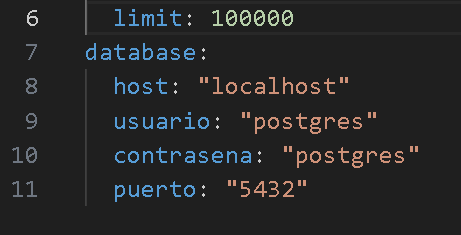




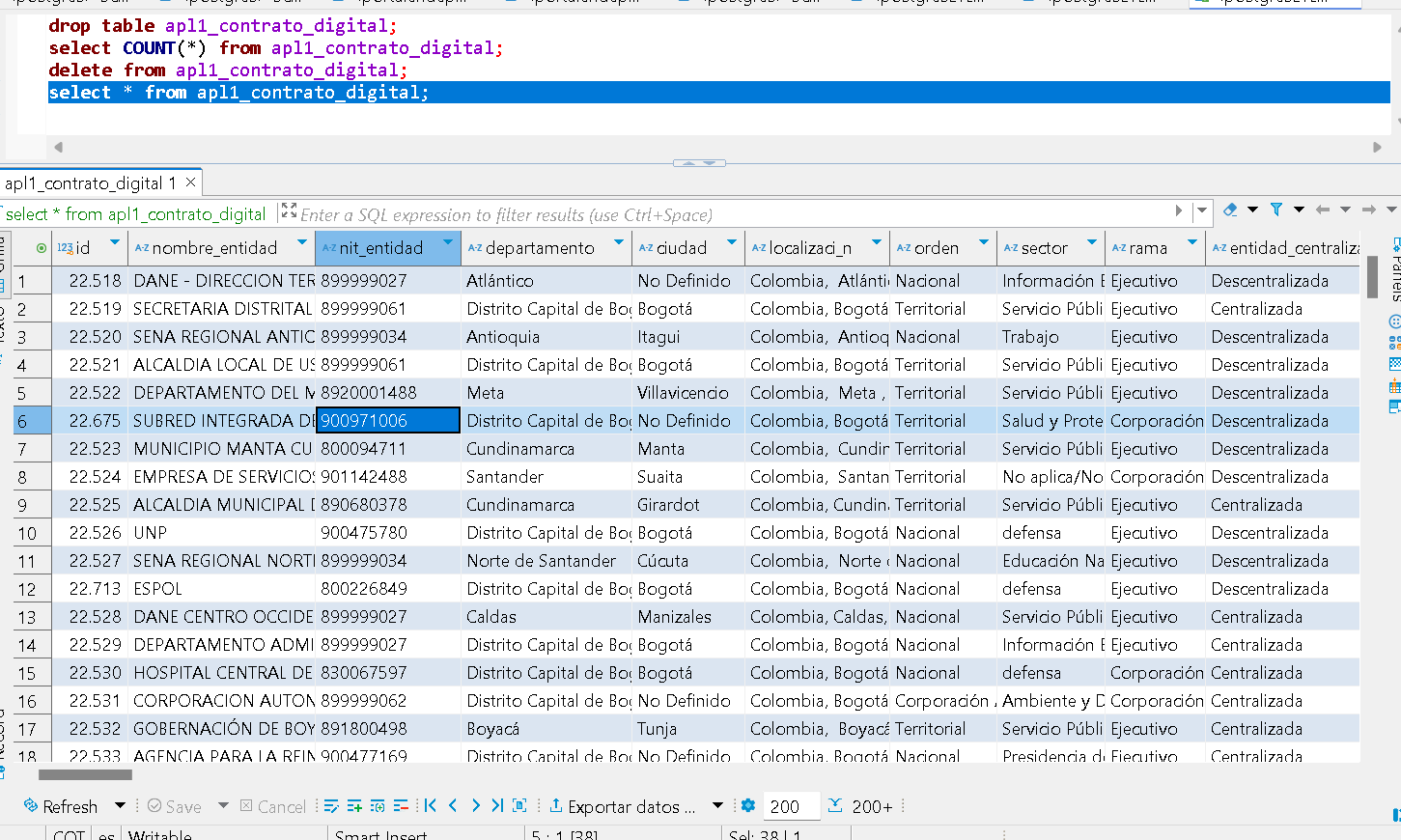
Se presenta errores con las fechas de los contratos, no registran para todos los contratos.



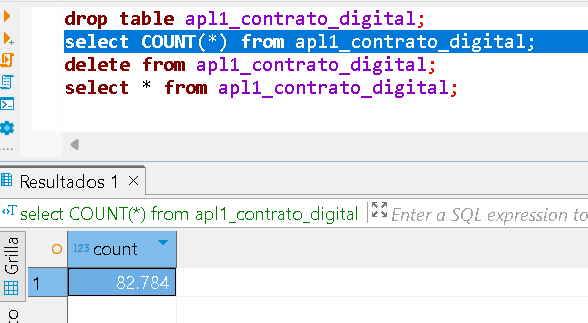
Se cambia el parámetro de 1000 por 100 mil registros y se procede a ejecutar el proceso.



Se verifica los datos insertados en la tabla apl1\_contrato\_digital en la base de datos de postgres.



Se realiza un select y se verifica que de 100 mil se lograron cargar correctamente 82784 registros.



Se sube el proyecto sin la base de datos de información por el tamaño limitado.

