

Proyecto Fundamentos Básicos de Python

Juan José Grajales Preciado
Juan Manuel Agudelo Lopez
Renzo Nicolás Usma Barrios
Docente: Alejandro Rodas Vásquez
Universidad Tecnológica de Pereira

2 de septiembre de 2024

Introducción

En este proyecto usted pondrá en práctica los conceptos básicos vistos en esta unidad especialmente *arreglos*, *funciones*, *módulos*. Se deberá consumir los recursos suministrados por una API suministrada por el portal Nacional de [Datos Abiertos](#) y crear una Arquitectura de Software basada en módulos.

Durante esta pandemia la adquisición, almacenamiento y consulta de datos relacionado con el reporte de casos de COVID-19 se ha convertido en un insumo imprescindible para la creación de aplicaciones software que ayuden en la prevención y trazabilidad de la enfermedad.

El portal de Datos Abierto pone a disposición el conjunto de datos llamado [Casos positivos de COVID-19 en Colombia](#) (se recomienda que sea examinado) el cual puede ser consultado por aplicaciones web o de escritorio a través de consultas a su API. A continuación, se encuentra el código fuente en Python que permite realiza dicha consulta.

```

1 import pandas as pd
2 from sodapy import Socrata
3
4 # Unauthenticated client only works with public data sets. Note 'None'
5 # in place of application token, and no username or password:
6 client = Socrata("www.datos.gov.co", None)
7
8 # Example authenticated client (needed for non-public datasets):
9 # client = Socrata(www.datos.gov.co,
10 #                 MyAppToken,
11 #                 username="user@example.com",
12 #                 password="AFakePassword")
13
14 # First 2000 results, returned as JSON from API / converted to Python
15 # list of dictionaries by sodapy.
16 results = client.get("gt2j-8ykr", limit=limite_registros, departamento =
17                     nombre_departamento)
18
19 # Convert to pandas DataFrame
20 results_df = pd.DataFrame.from_records(results)

```

En las líneas 1 y 2 se hace la llamada a las librerías *Pandas* y *sodapy*. *Pandas* es reconocida por ser empleada en *Análisis de Datos*. En la línea 16 se encuentran los parámetros *limite_registros* y *nombre_departamento*. Estos parámetros son requeridos y como su nombre lo dice, permiten consultar el Departamento y el Número de Casos que se quieren extraer. Para mayor conocimiento sobre el API se tiene el sitio [Documentación sodapy](#).

1. Requerimientos Funcionales

Usted debe de crear una aplicación que le permita al usuario ingresar el Departamento (*nombre_departamento*) que desea consultar y el Número de Registro (*limite_registros*) que quiere obtener de la consulta (este parámetro es importante puesto que si selecciona un número grande e.j 1000 registros, la consulta puede “colgarse”).

El resultado obtenido debe de poder visualizarse en la pantalla en un formato (investigar la función *format*) que contenga solo las siguientes columnas: *Ciudad de ubicación, Departamento, Edad, Tipo, Estado y País de procedencia*.

2. Requerimientos de la Arquitectura de Software

El software debe de estar construido aplicando el concepto de *modularidad*. Cada módulo debe de tener una tarea en específico

- Módulo para UI
- Módulo API
- Como archivo separado el *main*

3. Evidencias para la entrega

1. Usted debe de subir el código fuente a su repositorio invidual de [github](#).
2. Pantallazos donde se corrobore el funcionamiento del software (consultas realizadas y resultado obtenido) y se compruebe que los Requerimientos Funcionales ([Sección 1](#)) han sido cumplidos.
3. Para que esta actividad esté enmarcada dentro del contexto de la Arquitectura de Software y tengo un fundamento teórico válido. Usted debe de consultar ¿Qué es un Diagrama de Componentes? ¿Para qué se utiliza? ¿Qué es un componente?

3.1. Estructura del Proyecto

A continuación, se muestra la arquitectura que debe de tener el software ([Figura 1](#)). Sin embargo, en esta imagen solo se muestra la estructura base, se debe agregar los archivos Python que usted crea.

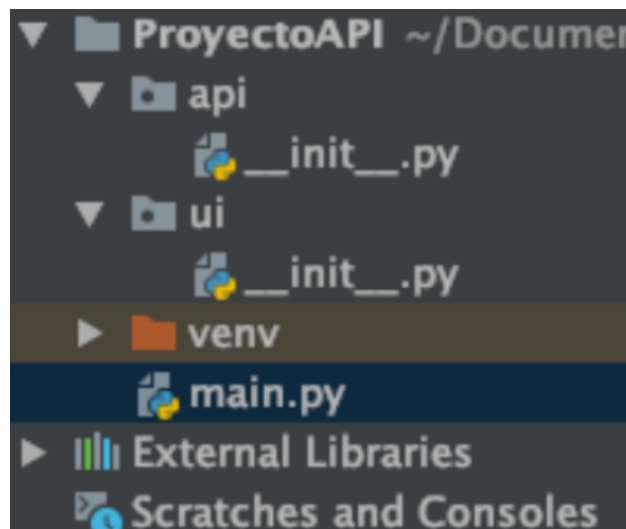
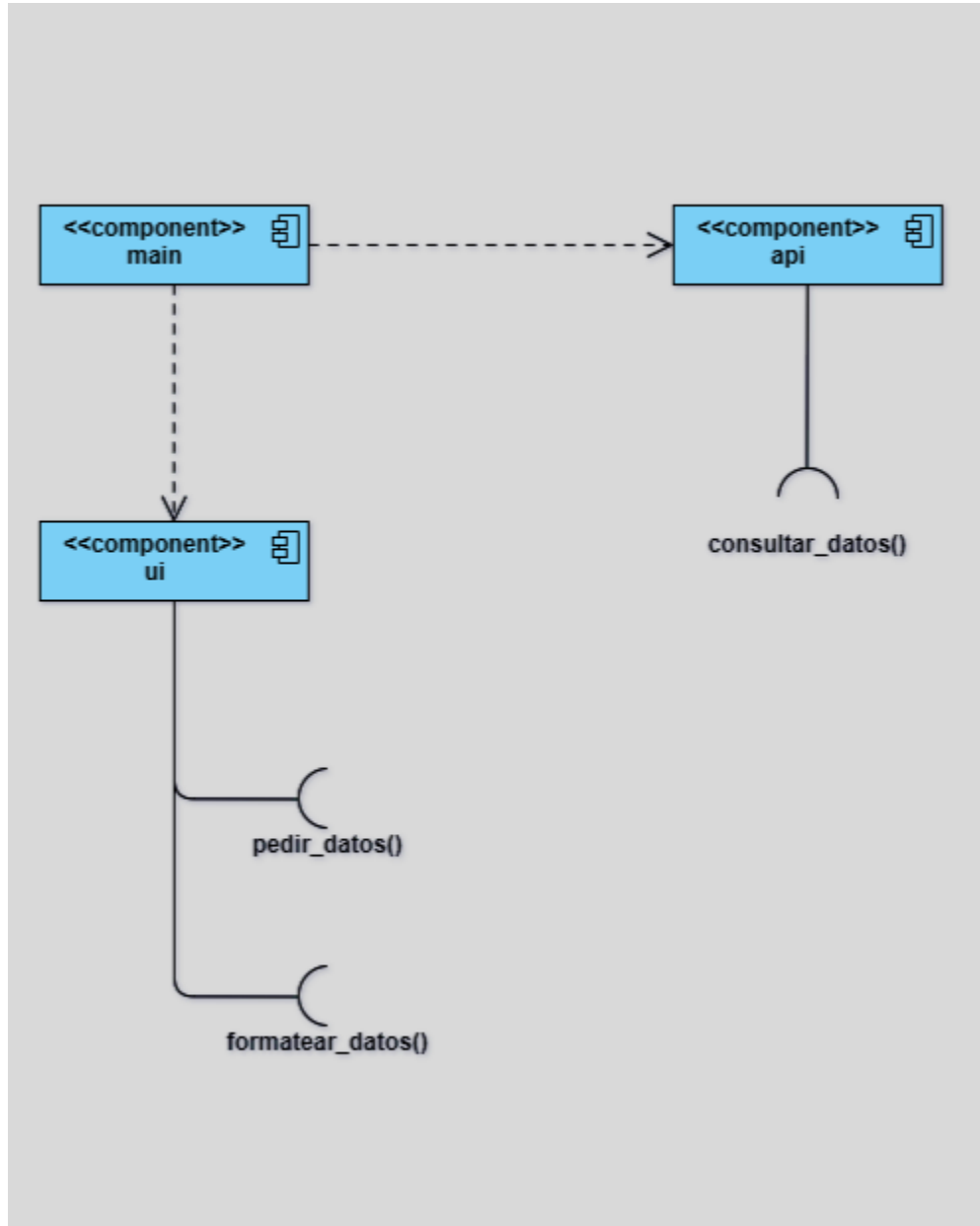


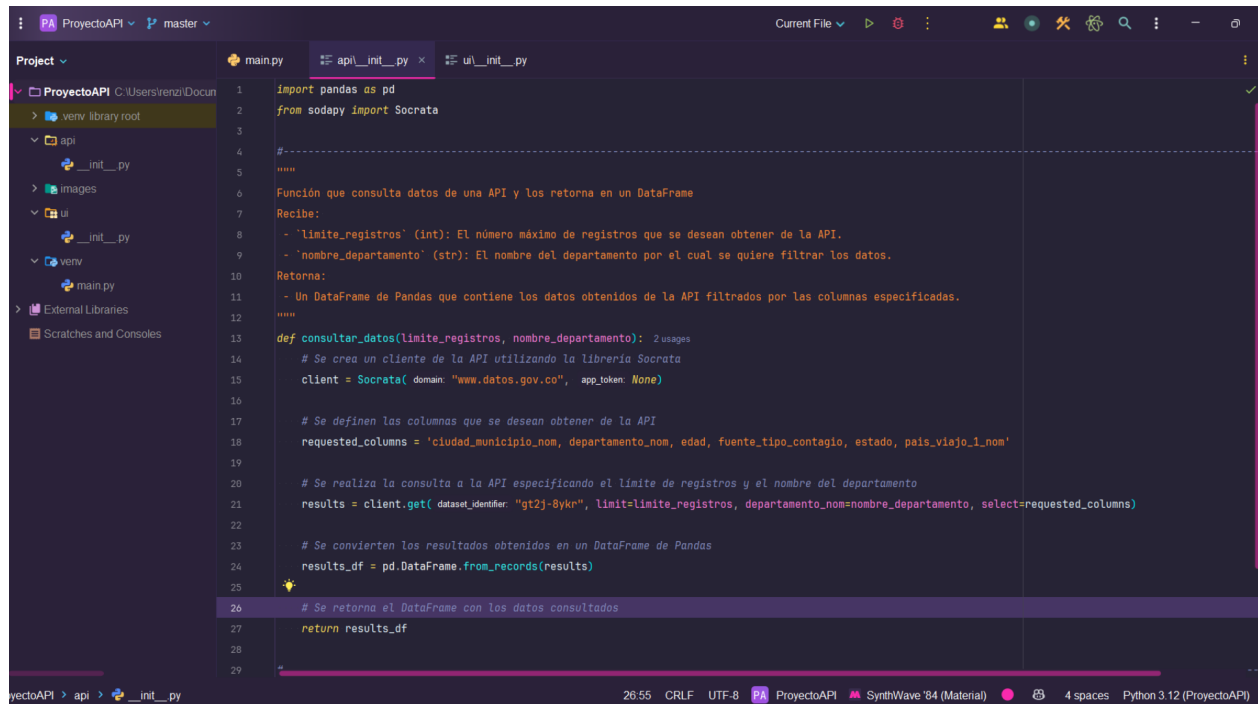
Figura 1: Estructura de la Arquitectura

3.2. Pantallazos

Evidencia 1. Diagrama de componentes



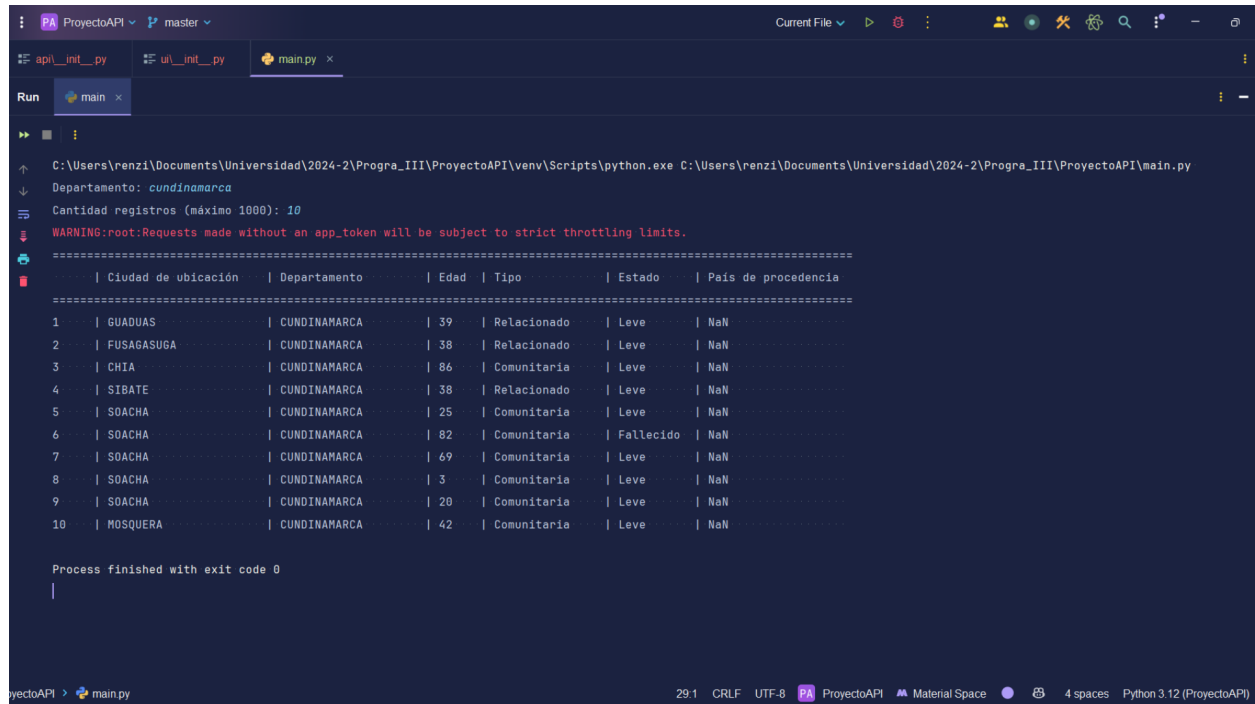
Evidencia 2. Requerimientos funcionales



The screenshot shows a VS Code editor with a project named 'ProyectoAPI'. The file explorer on the left shows the project structure, including a 'venv' directory and a 'main.py' file. The main editor displays the content of 'main.py', which is a Python script. The script imports 'pandas as pd' and 'Socrata' from 'sodapy'. It defines a function 'consultar_datos' that takes 'limite_registros' and 'nombre_departamento' as arguments. The function creates a Socrata client, defines requested columns, makes a GET request to the Socrata API, and returns a pandas DataFrame. The script is written in Spanish and includes comments in Spanish.

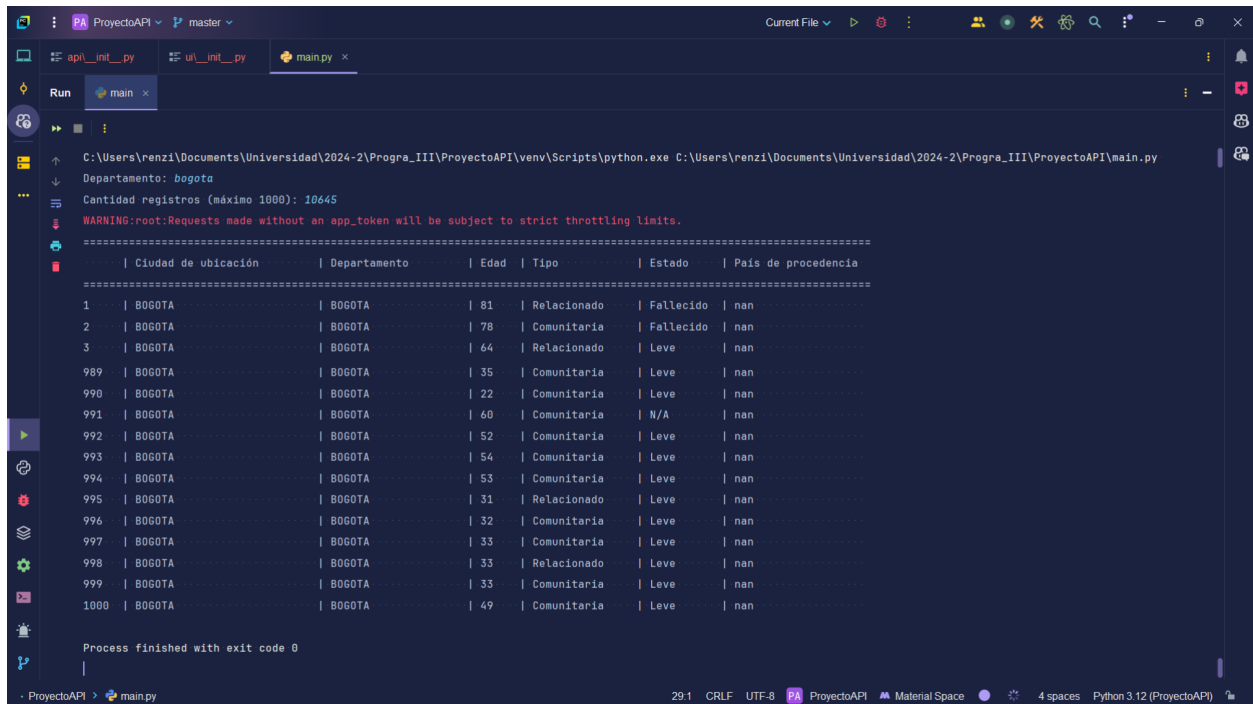
```
1 import pandas as pd
2 from sodapy import Socrata
3
4 #-----
5
6 Función que consulta datos de una API y los retorna en un DataFrame
7
8 Recibe:
9 - 'limite_registros' (int): El número máximo de registros que se desean obtener de la API.
10 - 'nombre_departamento' (str): El nombre del departamento por el cual se quiere filtrar los datos.
11
12 Retorna:
13 - Un DataFrame de Pandas que contiene los datos obtenidos de la API filtrados por las columnas especificadas.
14
15 def consultar_datos(limite_registros, nombre_departamento):
16     # Se crea un cliente de la API utilizando la libreria Socrata
17     client = Socrata( domain= "www.datos.gov.co", app_token= None)
18
19     # Se definen las columnas que se desean obtener de la API
20     requested_columns = 'ciudad_municipio_nom, departamento_nom, edad, fuente_tipo_contagio, estado, pais_viaje_1_nom'
21
22     # Se realiza la consulta a la API especificando el limite de registros y el nombre del departamento
23     results = client.get( dataset_idenfier= "gt2j-8ykr", limit=limite_registros, departamento_nom=nombre_departamento, select=requested_columns)
24
25     # Se convierten los resultados obtenidos en un DataFrame de Pandas
26     results_df = pd.DataFrame.from_records(results)
27
28     # Se retorna el DataFrame con los datos consultados
29     return results_df
```

Evidencia 3. Consultas realizadas



```
PA ProyectoAPI master Current File
api_init.py ui_init.py main.py x
Run main x
C:\Users\renzi\Documents\Universidad\2024-2\Progra_III\ProyectoAPI\venv\Scripts\python.exe C:\Users\renzi\Documents\Universidad\2024-2\Progra_III\ProyectoAPI\main.py
Departamento: cundinamarca
Cantidad registros (máximo 1000): 10
WARNING:root:Requests made without an app_token will be subject to strict throttling limits.
=====
| Ciudad de ubicación | Departamento | Edad | Tipo | Estado | Pais de procedencia |
=====
1 | GUADUAS | CUNDINAMARCA | 39 | Relacionado | Leve | NaN |
2 | FUSAGASUGA | CUNDINAMARCA | 38 | Relacionado | Leve | NaN |
3 | CHIA | CUNDINAMARCA | 86 | Comunitaria | Leve | NaN |
4 | SIBATE | CUNDINAMARCA | 38 | Relacionado | Leve | NaN |
5 | SOACHA | CUNDINAMARCA | 25 | Comunitaria | Leve | NaN |
6 | SOACHA | CUNDINAMARCA | 82 | Comunitaria | Fallecido | NaN |
7 | SOACHA | CUNDINAMARCA | 69 | Comunitaria | Leve | NaN |
8 | SOACHA | CUNDINAMARCA | 3 | Comunitaria | Leve | NaN |
9 | SOACHA | CUNDINAMARCA | 20 | Comunitaria | Leve | NaN |
10 | MOSQUERA | CUNDINAMARCA | 42 | Comunitaria | Leve | NaN |
=====
Process finished with exit code 0
|
ProyectoAPI > main.py 29.1 CRLF UTF-8 PA ProyectoAPI Material Space 4 spaces Python 3.12 (ProyectoAPI)
```

Evidencia 4. Límite máximo de registros mostrados



```
C:\Users\renzi\Documents\Universidad\2024-2\Progra_III\ProyectoAPI\venv\Scripts\python.exe C:\Users\renzi\Documents\Universidad\2024-2\Progra_III\ProyectoAPI\main.py
Departamento: bogota
Cantidad registros (máximo 1000): 10645
WARNING:root:Requests made without an app_token will be subject to strict throttling limits.
=====
| Ciudad de ubicación | Departamento | Edad | Tipo | Estado | País de procedencia |
=====
1 | BOGOTA | BOGOTA | 81 | Relacionado | Fallecido | nan
2 | BOGOTA | BOGOTA | 78 | Comunitaria | Fallecido | nan
3 | BOGOTA | BOGOTA | 64 | Relacionado | Leve | nan
989 | BOGOTA | BOGOTA | 35 | Comunitaria | Leve | nan
990 | BOGOTA | BOGOTA | 22 | Comunitaria | Leve | nan
991 | BOGOTA | BOGOTA | 60 | Comunitaria | N/A | nan
992 | BOGOTA | BOGOTA | 52 | Comunitaria | Leve | nan
993 | BOGOTA | BOGOTA | 54 | Comunitaria | Leve | nan
994 | BOGOTA | BOGOTA | 53 | Comunitaria | Leve | nan
995 | BOGOTA | BOGOTA | 31 | Relacionado | Leve | nan
996 | BOGOTA | BOGOTA | 32 | Comunitaria | Leve | nan
997 | BOGOTA | BOGOTA | 33 | Comunitaria | Leve | nan
998 | BOGOTA | BOGOTA | 33 | Relacionado | Leve | nan
999 | BOGOTA | BOGOTA | 33 | Comunitaria | Leve | nan
1000 | BOGOTA | BOGOTA | 49 | Comunitaria | Leve | nan
=====
Process finished with exit code 0
```

Evidencia 5. Validación de columna 'País de origen'



```
=====
Id | Ciudad de ubicación | Departamento | Edad | Tipo | Estado | País de procedencia
=====
1 | PALMIRA | VALLE | 43 | Importado | Leve | ESPAÑA
2 | PEREIRA | RISARALDA | 50 | Importado | Leve | ESPAÑA
3 | CHINCHINA | CALDAS | 46 | Importado | Leve | ESPAÑA
4 | MEDELLIN | ANTIOQUIA | 53 | Importado | Leve | ESPAÑA
5 | PEREIRA | RISARALDA | 38 | Importado | Leve | ESPAÑA
6 | PALMIRA | VALLE | 40 | Importado | Leve | ESPAÑA
7 | CARTAGO | VALLE | 30 | Importado | Leve | ESPAÑA
359 | BOLIVAR | VALLE | 66 | Importado | Leve | ESPAÑA
360 | CALI | VALLE | 27 | Importado | Leve | ESPAÑA
361 | CALI | VALLE | 44 | Importado | Leve | ESPAÑA
362 | CUCUTA | NORTE SANTANDER | 24 | Importado | Leve | ESPAÑA
363 | PALMIRA | VALLE | 47 | Importado | Leve | ESPAÑA
364 | CALI | VALLE | 40 | Importado | Leve | ESPAÑA
365 | CALI | VALLE | 23 | Importado | Leve | ESPAÑA
366 | CUCUTA | NORTE SANTANDER | 34 | Importado | Leve | ESPAÑA
367 | CALI | VALLE | 65 | Importado | Leve | ESPAÑA
368 | CALI | VALLE | 69 | Importado | Leve | ESPAÑA
369 | CALI | VALLE | 66 | Importado | Leve | ESPAÑA
370 | CALI | VALLE | 50 | Importado | Leve | ESPAÑA
=====

Process finished with exit code 0
```