

Caso Práctico

Ejercicio 1

He cambiado los errores que había en el código de sintaxis;

- El método para conectarse a la base de datos no es `con()`, sino `connect()`
- Para crear el cursor hay que llamar a la conexión y luego al cursor `self.conn.cursor()`
- El método no es `ex()`, es `execute()`
- `Executor()` tampoco lo es, hay que utilizar `executemany()` para la insertar más de una fila
- En lenguaje SQL, después de `FROM` falta el nombre de la tabla de donde se quiere extraer y después de `ORDER BY` poner el atributo según el que se quiere ordenar
- Finalmente, hay que cerrar la conexión con `self.conn.close()`

Tras esto y ejecutar el código:

```

Iniciando procesamiento de indicadores económicos...
✅ Datos insertados en la base de datos desde data/indicadores.

Últimos indicadores económicos insertados:
=====
Fecha: 1/2024 | Valor: 81.20 | Indicador: IVUS-211 | Insertado: 2025-02-20 11:41:28
Fecha: 12/2023 | Valor: 84.60 | Indicador: IVUS-211 | Insertado: 2025-02-20 11:41:28
Fecha: 11/2023 | Valor: 91.50 | Indicador: IVUS-211 | Insertado: 2025-02-20 11:41:28
Fecha: 10/2023 | Valor: 98.70 | Indicador: IVUS-211 | Insertado: 2025-02-20 11:41:28
Fecha: 9/2023 | Valor: 102.70 | Indicador: IVUS-211 | Insertado: 2025-02-20 11:41:28
Fecha: 8/2023 | Valor: 97.90 | Indicador: IVUS-211 | Insertado: 2025-02-20 11:41:28
Fecha: 7/2023 | Valor: 84.60 | Indicador: IVUS-211 | Insertado: 2025-02-20 11:41:28
Fecha: 6/2023 | Valor: 92.60 | Indicador: IVUS-211 | Insertado: 2025-02-20 11:41:28
Fecha: 5/2023 | Valor: 88.10 | Indicador: IVUS-211 | Insertado: 2025-02-20 11:41:28
Fecha: 4/2023 | Valor: 75.40 | Indicador: IVUS-211 | Insertado: 2025-02-20 11:41:28
=====
✅ Proceso finalizado.
(caso_practico_nte) (base) manuvillegas@MacBook-Pro-2 caso_practico %
```

(no me funcionaba la url primaria y he tenido que levantar el servidor http, pero he tenido que cambiar la dirección a 127.0.0.1 en vez de 127.0.0.42 ya que la local en mi ordenador era esa)

Ejercicio 2

Para este ejercicio he creado la función *opcional_pandas* en el archivo `sqlite_mineco_alumnos.py`;

1. Primero defino la variable `df` (que será el dataframe) leyendo de la base de datos que se pasa como parámetro y lee la tabla `indicadores_enconomicos`, con esos datos crea el dataframe
2. Después, los ordeno con el `sort.values` por fechas
3. En el dataframe aún tengo más valores a parte de la fecha y el valor que no nos interesan según la salida deseada, por lo tanto creo un nuevo dataframe a partir de el que teníamos desde el principio pero solo copiando las columnas 'fecha' y 'valor'
4. Hay muchos valores de un mismo mes, por lo tanto agrupo todos los registros en meses y me quedo con la media como valor, además, reseteo el índice
5. Tenemos un float de muchos decimales, pero según la salida esperada solo queremos dos decimales y el '%', por lo que aplico una función lambda para convertir `x` en `x` con dos decimales y el %
6. Finalmente la muestro para comprobar que todo se ha realizado de manera correcta.

He subido todo a un [repositorio de github](#) donde se ven todos los cambios realizados del caso en los commits