Base De Datos SQLite – Django

Andres Felipe Gonzalez Berdugo

Servicio Nacional De Aprendizaje

Centro Minero sede iteandes

Análisis y desarrollo de software (ADSO)

Ficha:2993648

Andrés Felipe Sandoval Higuera

Sogamoso, Boyacá

Noviembre 2025

Contenido

INTRODUCCION

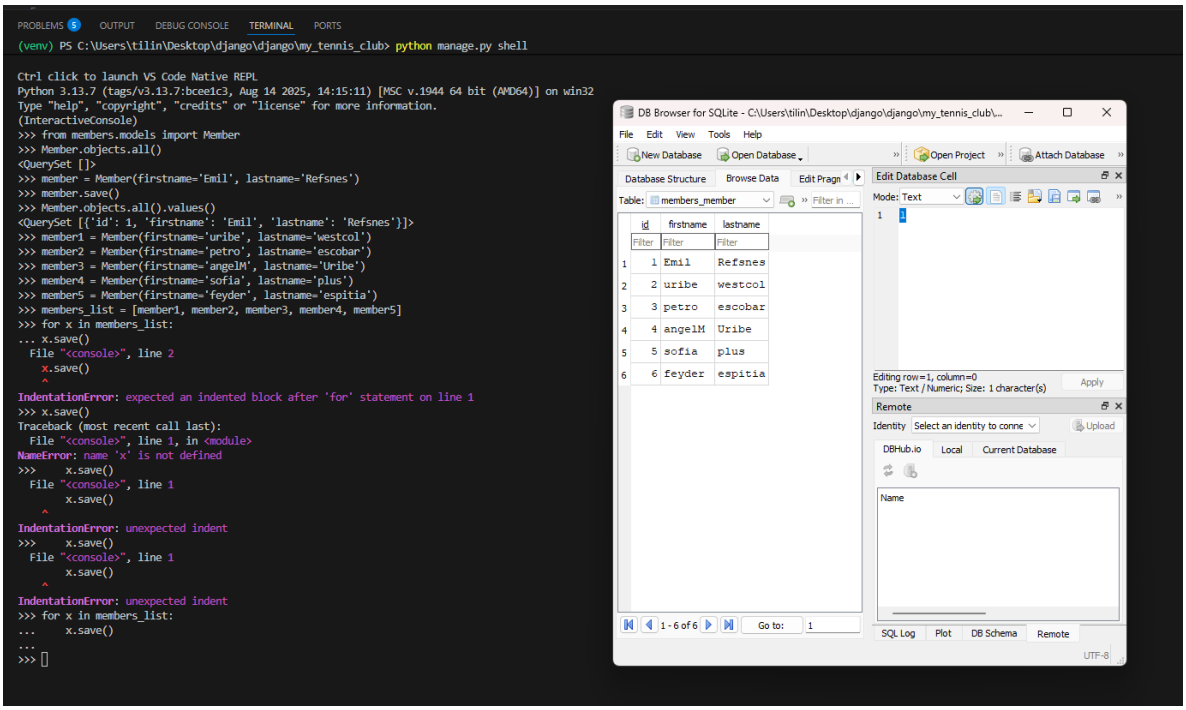
En este trabajo se dará a conocer todo lo visto en base de datos con el Framework Django usando el programa SQLite, mostrando en capturas de pantalla los comandos básicos vistos en clase.

Se verán varios puntos como los son la inserción de datos, consulta de datos, actualización de datos y la eliminación de los mismos.

Con este trabajo pretendo demostrar el buen uso y entendimiento de lo visto en clase.

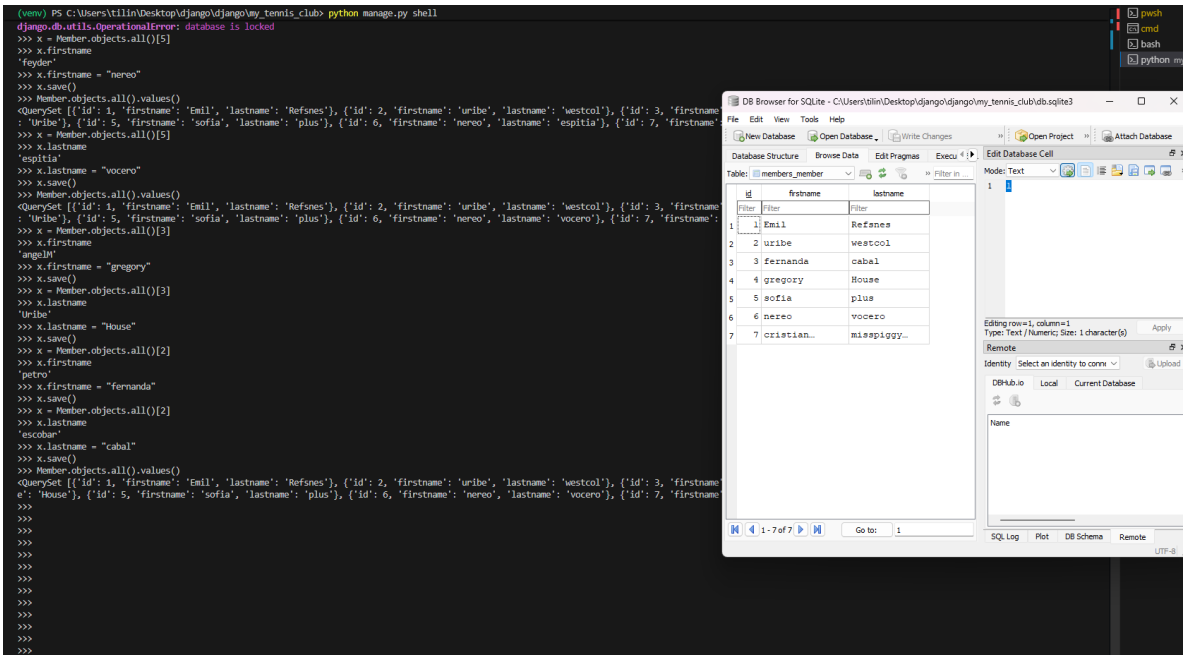
Base de datos SQLite Django

Inserción de datos



Se insertaron los datos a la tabla de datos previamente creada con “name” y “lastname”, esto se hizo mediante una variable “x” para insertar los datos

Actualización de datos



Por medio de los comandos que serán presentados en un momento se puede editar la información de las casillas designando a que usuario o en que casilla se van a hacer cambios

>>> from members.models import Member

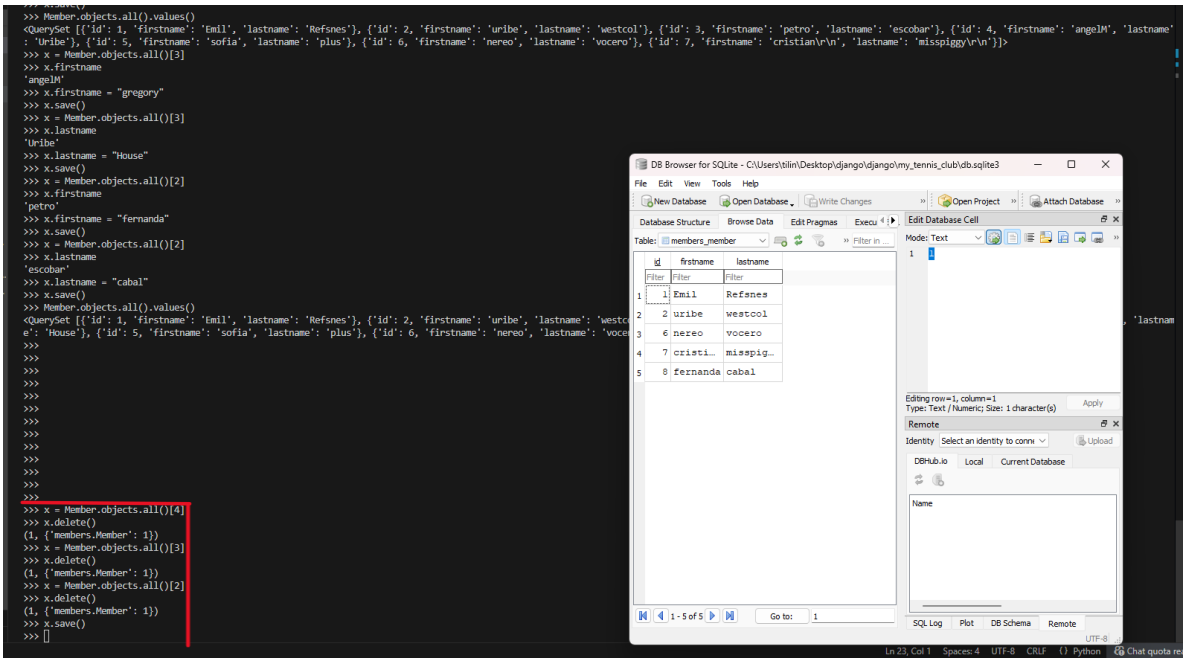
>>> Member.objects.all()

>>> x = Member.objects.all()[4]

>>> x.lastname

>>> x.lastname = "cabal"

Eliminar datos



Como se puede evidenciar en el recuadro rojo se uso el comando “x.delete() el comando es similar al de editar caracteres

>>> x = Member.objects.all()[4]

>>> x.delete()

Ejercicio 1

Se tienen que agregar 2 nuevas columnas las cuales serán el teléfono y fecha de entrada las cuales se agregaran mediante código en el archivo “models.py” como lo podemos apreciar en la pagina w3schools compartida en clase, luego de hacer los cambios en el código de “models.py” tendremos que realizar las migraciones a la base de datos para que esta se actualice

my_tennis_club/members/models.py:

```
from django.db import models

class Member(models.Model):
  firstname = models.CharField(max_length=255)
  lastname = models.CharField(max_length=255)
  phone = models.IntegerField()
  joined_date = models.DateField()
```

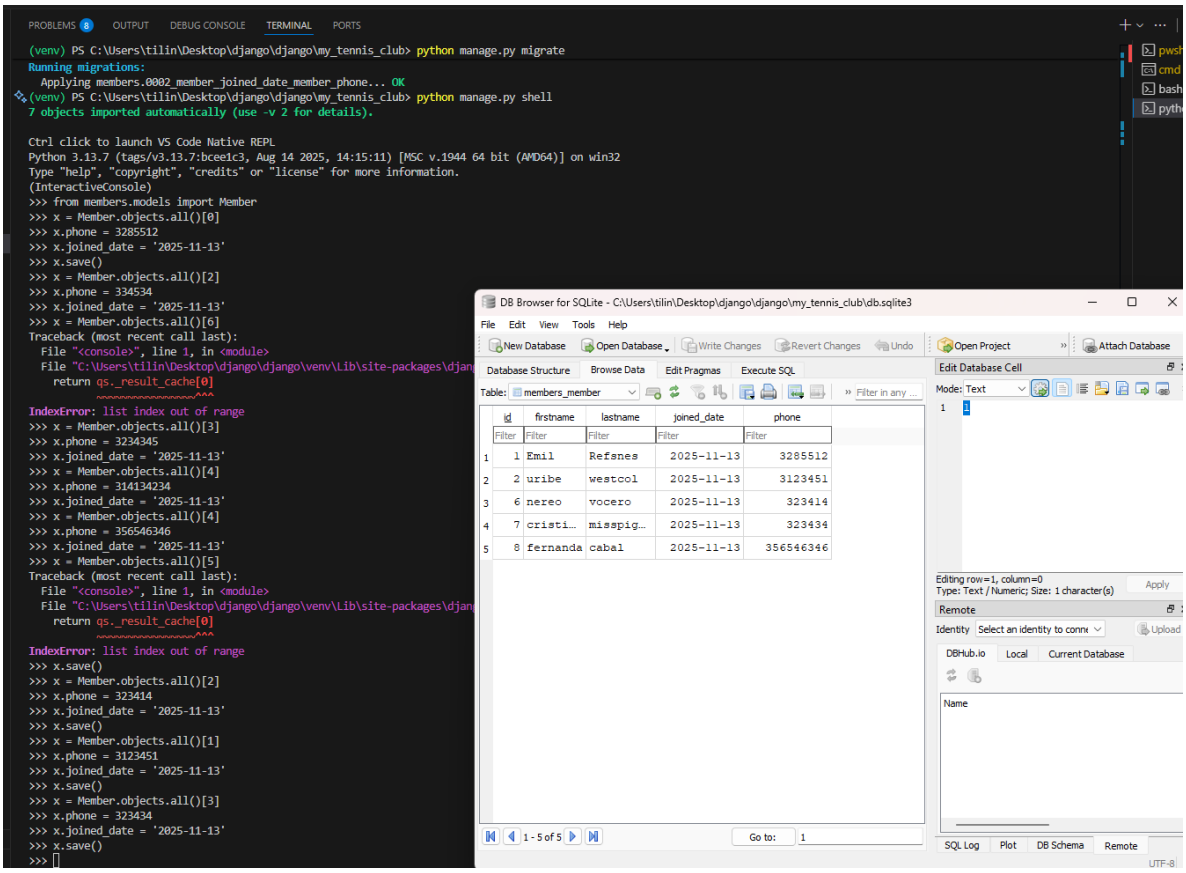
As you can see, we want to add `phone` and `joined_date` to our Member Model.

This is a change in the Model's structure, and therefor we have to make a migration to tell Django that it has to update the database:

```
python manage.py makemigrations members
```

Luego se usara el código Python manage.py migrate con el cual sabremos si migraron de manera correcta las columnas anteriormente creadas

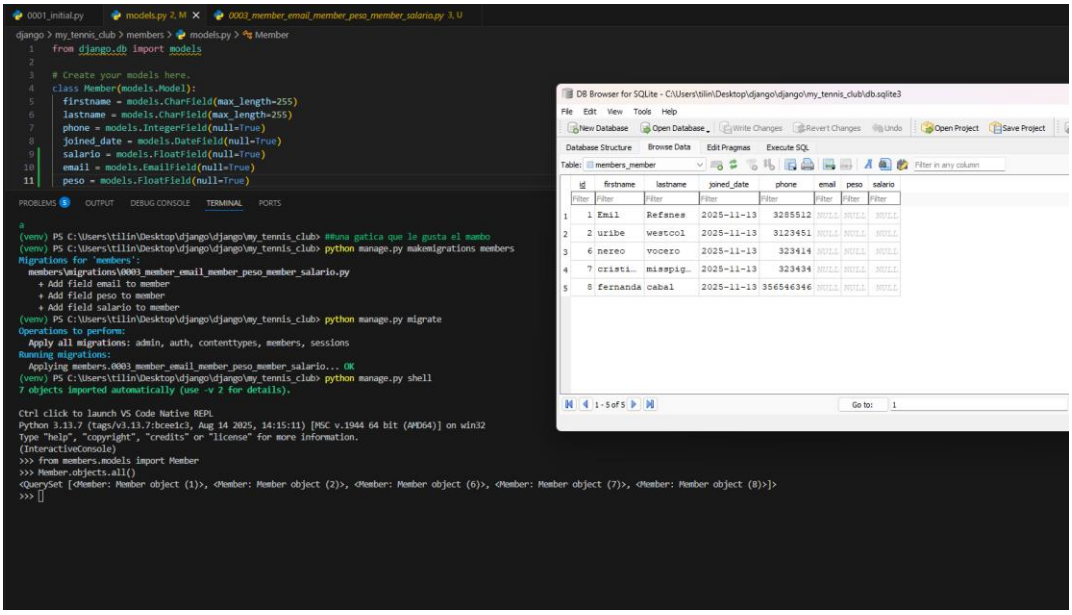
Una vez hecho esto se proceden a agregar los datos de los usuarios a las 2 nuevas columnas, en mi caso en el código se aprecian errores por mala sintaxis al momento de realizarse, sintaxis que corregí allí mismo



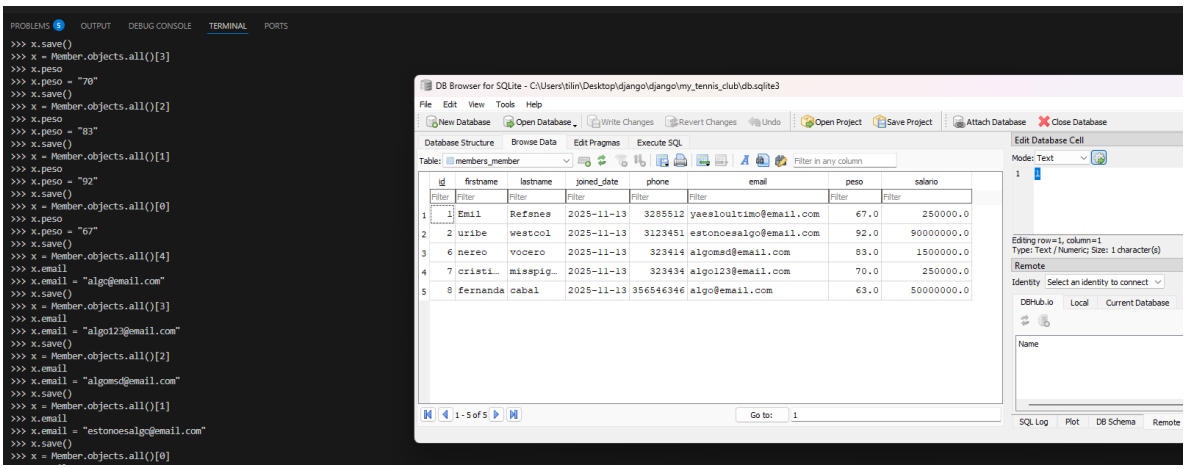
Como resultado final se logra evidenciar en la tabla de datos que se ingresaron correctamente los datos en cada columna

Ejercicio 2

En este caso debemos agregar 3 nuevas columnas a la base de datos, entonces se hará de la manera explicada antes, son prácticamente los mismos códigos lo único que cambia en este caso son nombres y variables ya que por mi parte los nombres de las nuevas columnas a las cuales voy a a agregar información en la base de datos son (salario, email y peso)



Aquí podemos evidenciar como se crearon las nuevas columnas para guardar información nueva, como dicho anteriormente (peso, email y salario)



Como se logra evidenciar se agregaron los datos a sus respectivas columnas de su respectivo usuario

CONCLUSIONES

-Con este proyecto se pudo evidenciar un buen manejo de la plataforma Django para la gestión del proyecto

-se demostró buena practicidad en la manipulación y manejo de la base de datos

-Este proyecto funciona mucho para aprender los temas básicos de bases de datos con el Framework Django