Managers in django

Castro Soberanes Luis Alexis

Universidad Tecnológica de Tijuana.
TSU. Tecnologías de la información.
Entornos virtuales y negocios digitales.
Docente:Parra Galaviz Ray Brunett

Introduction

In the following document, you will delve into a detailed investigation into the use of managers in Django, a fundamental topic in web application development. We will explore in depth what they are, how they are used, their essential characteristics and a concrete example of a type of manager. Our intention is to provide you with a solid understanding of this crucial topic in Django. Additionally, we have included an illustrative image to enrich your learning experience.

What are Managers

A manager in Django is a class that provides methods for performing queries and operations on the database in relation to a specific model. Each Django model has at least one default manager called objects, which allows you to interact with the database in a simple way. [1]

Creating a Custom Manager

To create a custom manager, you need to define a class that inherits from models.Manager and add custom methods to perform the operations you desire. Then, assign this class as an attribute in your model [1]

Custom managers

Custom managers in Django are managers that you define in your models to add specific functionality to database management. Although all models in Django have at least one default manager called "objects," you can create your custom managers to perform custom queries, apply specific filters, or carry out other database operations according to your application's needs. [1]

What are they used for [2]

Managers are used for the following tasks:

- Performing database queries to retrieve records that match certain criteria.
- Creating new records in the database.
- Updating existing records.
- Deleting records from the database.
- Performing aggregation and data analysis operations.

Features of managers in Django [2]

- Managers provide predefined methods such as ‘filter()’, ‘get()’, ‘create()’, ‘all()', and more to interact with the database.
- You can define your custom managers in a model if you want to add specific methods for that model.
- Managers allow you to perform complex database queries through method chaining, making it easier to construct complex SQL queries without writing SQL directly.

Conclusion

From my point of view I came to the conclusion that "administrators" in Django make interacting with the database easier. These managers allow operations such as queries, insertions, updates and deletions of records in the database to be carried out in an efficient and organized manner. Additionally, custom managers provide the flexibility to tailor the models' functionality to the specific needs of the application, resulting in cleaner, more maintainable code. In short, administrators are an essential tool for working with databases in Django applications.

BIBLIOGRAFIAS IEEE

[1] [En línea]. Available: https://docs.djangoproject.com/en/4.2/topics/db/managers/.

[2] [En línea]. Available: https://coffeebytes.dev/managers-o-manejadores-personalizados-en-django/.