# Django Models

A Django model is a table in your database.Django allows us to work with data, without having to change or upload files in the prosess.

In Django, data is created in objects, called Models, and is actually tables in a database.

## Create Table (Model)

1.Import Django models Module:

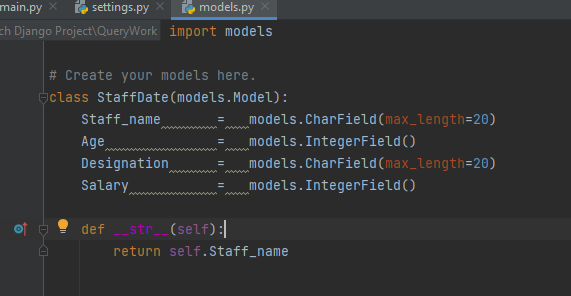
from django.db import models

Import the models module from Django, which provides the tools for defining database models.

2. class StaffDate(models.Model):

Define a new Django model named StaffDate that inherits from models.Model.

3. Define Model Fields:



* Define fields for the model using various field types provided by Django's models module.
* CharField is for storing character data (e.g., names and designations).
* IntegerField is for storing integer data (e.g., age and salary).
* max\_length specifies the maximum length for character fields.

4.

Define \_\_str\_\_ Method:

* Define a \_\_str\_\_ method to provide a human-readable string representation of instances of the model.
* In this case, it returns the Staff\_name, which helps when displaying instances in the Django admin or debugging.

This model represents a staff member with fields for their name, age, designation, and salary. You can use Django's migration system (python manage.py makemigrations and python manage.py migrate) to create the corresponding database table based on this model. Once the migrations are applied, you can use this model to interact with the database

5. Running the python manage.py makemigrations command in a Django project is the first step to create database migrations based on the changes made to your models. Here's how you would typically use this command:

Ensure your models are defined:

* 1.Before running makemigrations, make sure you have defined your models. In your case, you should have the StaffDate model in one of your Django apps.

2.Navigate to your project directory:

.Open a terminal or command prompt and navigate to the root directory of your Django project.

3.Run the makemigrations command:

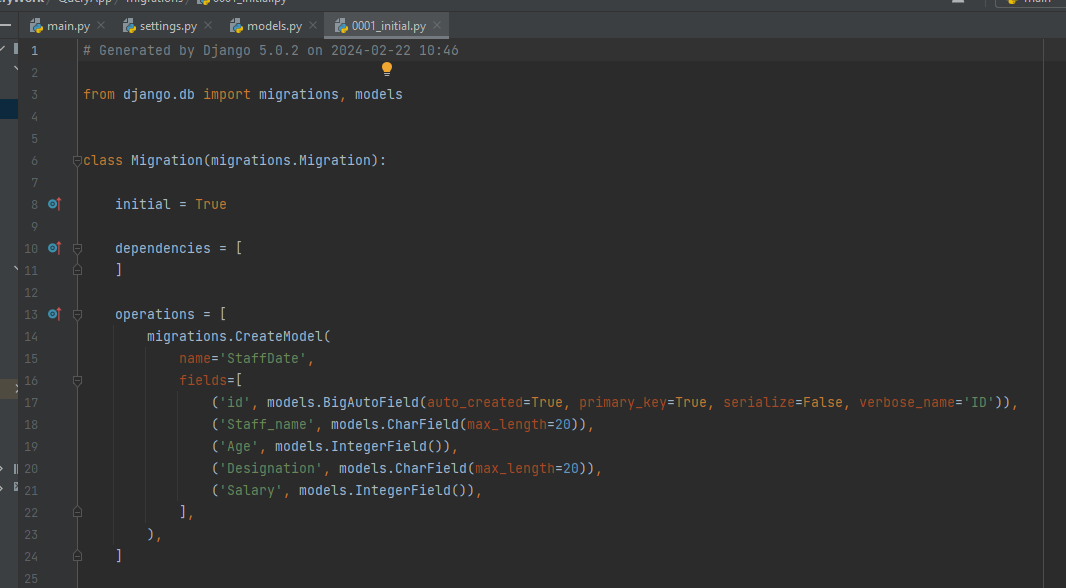
* python manage.py makemigrations

This command analyzes the models in your Django apps and creates new migration files in the migrations directory. These files describe the changes to be made to the database schema.

4.Review the migration files:

After running makemigrations, you can check the migrations directory to see the generated migration files. They are usually named with a timestamp and describe the changes to your database schema.

Django creates a file describing the changes and stores the file in the /migrations/ folder:



5.The table is not created yet, you will have to run one more command, then Django will create and execute an SQL statement, based on the content of the new file in the /migrations/ folder.

Run the migrate command:

Once you are satisfied with the generated migration files, apply the changes to the database using the migrate command:

* python manage.py migrate

By running these commands, you ensure that your database schema reflects the current state of your models. If you make further changes to your models in the future, you'll need to run makemigrations again to generate new migration files and then run migrate to apply those changes to the database.

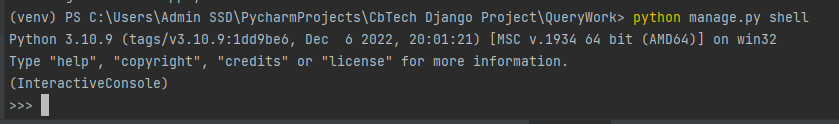
# Django Insert Data

In Django, you can use the Python shell to insert data into your model's table. Here's a step-by-step guide on how to insert data into the StaffDate model using the Django shell:

1)Open the Python shell in your Django project:To open a Python shell, type this command:

python manage.py shell

2)Now we are in the shell, the result should be something like this:

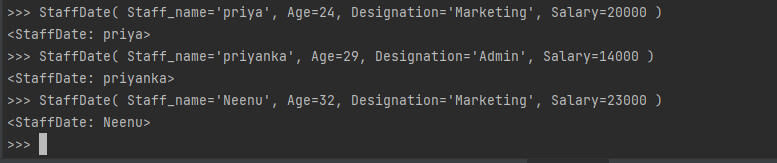


3)Import the StaffDate model:

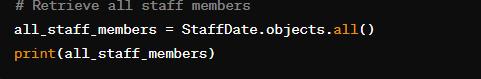
from QueryApp.models import StaffDate

Create a new instance of the StaffDate model and save it to the database:

4)Create a new instance of the StaffDate model and save it to the database:



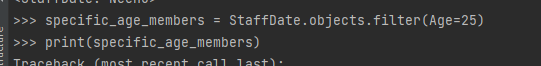
5)Verify the insertion by retrieving all staff members:



You can use the filter() method in Django to retrieve a subset of records from your database based on specified conditions. Here are some examples of how to use the filter() method:

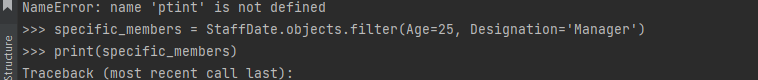
### **Basic Filtering:**

1)Filter by a Single Field:



2)

Filter by Multiple Fields:



3)Filter with Range:



### **Chain Multiple Filters:**

4)Chaining Filters:



5)**Combining Filtering and Ordering:**

Filter and Order By:



6)To filter StaffDate records where the Staff\_name starts with the letter 'A', you can use the startswith lookup in Django.



7)

To filter StaffDate records where the Salary is greater than 15000, you can use the gt (greater than) lookup in Django.



This will retrieve all StaffDate records where the Salary is greater than 15000. Adjust the conditions as needed for your specific requirements.