Модели (часть 1)

№ урока:	5	**Курс:**	Django Starter

Средства обучения:	Персональный компьютер с установленными: Python 3.8.2 Django 3.0.4

Обзор, цель и назначение урока

Цель данного урока - ознакомиться с основами работы Django приложения с базой данных. Научиться создавать собственные модели, настраивать их и научиться работать с ними. Также в процессе урока будет рассмотрено то, как подключить к приложению более сложную (Postgres) базу данных, чем имеющаяся в Django. Дополнительно будет рассмотрено взаимодействие между моделями или таблицами в базе данных.

Изучив материал данного занятия, учащийся сможет:

- Создавать собственные модели с правильными полями.
- Создавать взаимоотношения между моделями.
- Научиться подключать к Django проекту другие базы данных, например Postgres.

Содержание урока

1) Сравнение базы данных Postgres и SQLlite. Подключение кастомной базы данных к Django
2) Создание новых моделей и рассмотрение их пользы
3) Поля моделей. Типы полей и нюансы в работе с ними
4) Взаимодействие между моделями: one-to-one, many-to-many, foreignkey, one-to-many

Резюме

- **Модели** — это представление данных в качестве классов и их свойств. Каждая модель представляет собой класс Python, который является подклассом django.db.models.Model. Атрибут модели представляет поле базы данных.
- Поля. Самая важная часть модели (и единственная необходимая часть модели) — это список полей базы данных, которые она определяет. Поля определяются атрибутами класса.
- Каждое поле в модели должно быть экземпляром соответствующего класса Field. Django использует типы классов полей для определения нескольких вещей:
    - Тип столбца, который сообщает базе данных, какой тип данных хранить (например, INTEGER, VARCHAR, TEXT).
    - HTML-код по умолчанию widget, используемый при визуализации поля формы (например, <input type = "text">, <select>).
    - Минимальные требования проверки, используемые в админке Django и в автоматически сгенерированных формах.
- Основные типы полей:
    - IntegerField, целое число. Значения от -2147483648 до 2147483647 безопасны во всех базах данных, поддерживаемых Django.
    - BooleanField, поле истина/ложь.
    - CharField, строковое поле, для строк малого и большого размера. Для больших объемов текста используйте TextField.

CyberBionic systematics

CyberBionic Systematics ® 2020
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

    - DateField, дата, представленная в Python экземпляром datetime.date. Имеет несколько дополнительных необязательных аргументов.
    - DateTimeField, дата и время, представленные в Python экземпляром datetime.datetime. Принимает те же дополнительные аргументы, что и DateField.
    - DecimalField, десятичное число с фиксированной точностью, представленное в Python экземпляром Decimal. Он проверяет ввод с помощью DecimalValidator.
    - DurationField, поле для хранения периодов времени - смоделировано в Python с помощью timedelta. При использовании в PostgreSQL используемый тип данных представляет собой interval, а в Oracle тип данных представляет собой INTERVAL DAY (9) TO SECOND (6). В противном случае используется bigint микросекунд.
    - EmailField, класс CharField, который проверяет, является ли значение действительным адресом электронной почты, используя EmailValidator.
    - FileField, поле для загрузки файла.
    - FloatField, число с плавающей точкой, представленное в Python экземпляром float.
    - ImageField, наследует все атрибуты и методы из FileField, но также проверяет, что загруженный объект является допустимым изображением.
    - TextField, большое текстовое поле.
    - URLField, CharField для URL, проверяется валидатором URLValidator.
- Поля отношений:
    - ForeignKey, отношения многие-к-одному. Требуются два позиционных аргумента: класс, к которому относится модель, и опция on_delete.
    - ManyToManyField, отношения многие ко многим. Требуется позиционный аргумент: класс, к которому относится модель, который работает точно так же, как и для ForeignKey, включая рекурсивные и ленивые отношения. Связанные объекты можно добавлять, удалять или создавать с помощью поля RelatedManager.
    - OneToOneField, отношения один-к-одному. Концептуально это похоже на ForeignKey с unique=True, но «обратная» сторона отношения будет напрямую возвращать один объект.
- Опции полей. Следующие аргументы доступны для всех типов полей. Все необязательные.
    - null - если True, Django будет хранить пустые значения как NULL в базе данных. По умолчанию установлено значение False.
    - blank - если True, поле может быть пустым. По умолчанию установлено значение False.
    - choices (первый элемент в каждом кортеже) — это фактическое значение, которое должно быть установлено в модели, а второй элемент - удобочитаемое имя.
    - db_column - имя столбца базы данных для использования в этом поле. Если это не указано, Django будет использовать имя поля.
    - db_tablespace - имя табличного пространства базы данных, которое будет использоваться для индекса этого поля, если это поле проиндексировано.
    - default - значение по умолчанию для поля. Это может быть значение или вызываемый объект. Если он вызывается, он будет вызываться каждый раз, когда создается новый объект.
    - editable - если False, поле не будет отображаться ни админкой, ни какими-либо другими ModelForm. Они также пропускаются во время проверки модели. По умолчанию установлено значение True.
    - error_messages - аргумент error_messages позволяет вам переопределить сообщения по умолчанию, которые вызовет поле. Передайте словарь с ключами, соответствующими сообщениям об ошибках, которые вы хотите переопределить.
    - help_text - дополнительный «справочный» текст для отображения с виджетом формы. Это полезно для документации, даже если ваше поле не используется в форме.
    - primary_key - если True, это поле является первичным ключом для модели.

CyberBionic systematics
CyberBionic Systematics ® 2020
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

- unique - если True, это поле должно быть уникальным во всей таблице.
- unique_for_date - задайте для него имя DateField или DateTimeField, чтобы требовать, чтобы это поле было уникальным для значения поля даты.
- unique_for_month - как unique_for_date, но требует, чтобы поле было уникальным по отношению к месяцу.
- unique_for_year - как unique_for_date и unique_for_month.
- verbose_name - удобочитаемое имя для поля. Если подробное имя не указано, Django автоматически создаст его, используя имя атрибута поля, преобразовав подчеркивание в пробелы.
- validators - список валидаторов для этого поля.

Закрепление материала

- Что такое model (модели)?
- Какие есть типы полей?
- Какие могут быть отношения между моделями?
- Какие есть основные опции полей моделей?
- Какие действия нужно проделать чтобы подключить базу данных к Django приложению.

Дополнительное задание

Задание
Создать модели для интернет-магазина, которые будут отвечать следующим критериям:

- Модель должна вмещать все перечисленные в уроке поля.
- Добавить связь между моделями.
- Все поля по умолчанию должны быть пустыми.
- Добавить модели к как минимум 5 продуктам.

Самостоятельная деятельность учащегося

Задание 1
Изучить и понять все преимущества и недостатки инструментов, которые были рассмотрены на уроке.

Задание 2
Создать несколько моделей и реализовать между ними отношение много-к-много и попробовать запустить миграции.

Задание 3
Создать по 5 записей в базе данных для каждой модели.

Рекомендуемые ресурсы

Официальная документация Django:
https://www.djangoproject.com/

PostgreSQL документация:
https://www.postgresql.org/download/windows/

CyberBionic systematics
CyberBionic Systematics ® 2020
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com