Objekt

hinzufügen, löschen, updaten

python manage.py shell

Das Model Company

```
class Company(models.Model):


  COMPANY_TYPES = (
      ("tech", "food"),
  )
  name = models.CharField(max_length=100)
  description = models.TextField('Beschreibung der Firma')
  number_of_employees = models.IntegerField()
  company_type = models.CharField(
      max_length=10,
      choices=COMPANY_TYPES
  )
  sub_title = models.CharField(max_length=30, null=True, blank=True)
```

zu Erinnerung: das Model Company

neues Objekt anlegen

Um ein neues Objekt in der Datenbank anzulegen (in der Django shell oder in einer View), erstellen wir einfach ein neues Objekt der gewünschten Klasse.

```
from my_first_app.models import Company

company_1 = Company(name='Superduper AG‘, description='IT Firma aus Hamburg',
number_of_employees=5, company_type='tech')
```

mit der Methode save() des Models können wir das Objekt persistent in der Datenbank speichern:

```
company_1.save()
```

Objekt untersuchen

Das neu erstellte Objekt c ist ein Objekt der Klasse Company. Wir können auf die Attribute und Methoden dieses Objektes zugreifen.

company_1.name

Superduper Ag

dict

mit dem magic Attribut dict seine aktuellen Attribute untersuchen

company_1.__dict__

{

'_state': <django.db.models.base.ModelState object at 0x7fd1207f9160>,

'id': 1,

'name': 'Superduper AG',

'description': 'eine IT Firma aus Hamburg',

'number_of_employees': 5,

'company_type': 'tech',

'sub_title': None

}

id ist ein Autoincrement und wurde automatisch gesetzt. _state ist eine Klasse zum Halten von Instanz-Daten

Ein Objekt laden

Über die ID (=Primary Key) können wir ein Objekt laden. Dazu nutzen wir die get-Methode des Objekt-Managers des Models.

```
id=1
c = Company.objects.get(pk=id)
c
<Company: Suberduper AG>
```

Model Manager

Ein Manager ist die Schnittstelle, über die Datenbank-Operationen für ein Model bereitgestellt werden. Für jedes Model existiert zumindest ein Manager (default). Der name des Default-Managers ist objects.

```
>>Company.objects
<django.db.models.manager.Manager object at 0x7f8f09cb71c0>
```

Manager get Methode

Die get-Methode haben wir schon gesehen. Wir können ihr noch weitere Keyword-Argumente übergeben.

```
c = Company.objects.get(name='Superduper AG', number_of_employees=5)
c
<Company: Suberduper AG>
```

Wird auf Basis der Keyword Argumente kein Objekt gefunden, wird eine Exception ausgelöst.

Objekt löschen

Um ein Objekt aus der DB zu löschen, müssen wir die Objektinstanz erst über get() laden. Dann führen wir auf der Instanz die Methode delete() aus.

```
o = Company.objects.get(pk=1)
deleted_object = o.delete()
```

Das Objekt wurde jetzt aus der Datenbank gelöscht.

In deleted_object findet sich ein Tupel mit der ID des gelöschten Objektes.

Objekt updaten

Um ein Objekt (in der Datenbank) abzudaten, müssen wir die Objektinstanz erst über get() laden. Dann weisen wir ihr neue Werte zu und nutzen wieder die save()-Methode der Instanz selbst.

```
o = Company.objects.get(pk=1)
o.number_of_employees = 3
o.save()
```

Das Objekt wurde jetzt in der Datenbank gespeichert.