**ORM DATABASE RELATIONSHIP IN DJANGO USING SHELL**

**Many to Many Relationship Table Design**

**Class 1:**

class Programmer(models.Model):  
 name = models.CharField(max\_length=30)  
 salary = models.IntegerField(default=0)

**Class 2:**

class Projects(models.Model):  
 projectname = models.CharField(max\_length=30)  
 programmers = models.ManyToManyField(Programmer)

**Migrate**

~\django\_projects\fbvEmployees git:[master]

py -m manage makemigrations

Migrations for 'fbvEmpApp':

fbvEmpApp\migrations\0002\_programmer\_projects.py

- Create model Programmer

- Create model Projects

~\django\_projects\fbvEmployees git:[master]

py -m manage migrate

Operations to perform:

Apply all migrations: admin, auth, contenttypes, fbvEmpApp, sessions

Running migrations:

Applying fbvEmpApp.0002\_programmer\_projects... OK

In the command prompt go to the shell using :

~\django\_projects\fbvEmployees git:[master]

py -m manage shell

Python 3.12.3 (tags/v3.12.3:f6650f9, Apr 9 2024, 14:05:25) [MSC v.1938 64 bit (AMD64)] on win32

Type "help", "copyright", "credits" or "license" for more information.

(InteractiveConsole)

>>> from fbvEmpApp.models import Programmer, Projects

>>> prog1 = Programmer(name='Kamal Kumar', salary = 105000)

>>> prog1.save()

>>> prog2 = Programmer(name='Kingsly', salary = 135000)

>>> prog2.save()

>>> proj1 = Projects(projectname = 'CEM Tool - Python Project')

>>> proj1.save()

>>> proj2 = Projects(projectname = 'Security - Java Project')

>>> proj2.save()

>>> proj1.programmers.add(prog1)

>>> proj1.save()

>>> proj2.programmers.add(prog2)

>>> proj2.save()

>>>

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

In the class (models.py), we haven’t defined any relationships. To set it up, Django has an option of set, which we will do in the command prompt. (shell)

>>> prog1.projects\_set.all()

<QuerySet [<Projects: Projects object (1)>]>

>>> prog2.projects\_set.all()

<QuerySet [<Projects: Projects object (2)>]>

>>>

A screenshot of a computer

Description automatically generated

**Many to One Relationship Table Design (Foreign Key)**

**Class:**

class Customer(models.Model):  
 name = models.CharField(max\_length=30)  
  
  
class ContactDetails(models.Model):  
 type = models.CharField(max\_length=30)  
 tvalue = models.CharField(max\_length=30)  
 customer = models.ForeignKey(Customer, on\_delete=models.CASCADE)

**Migrate**

~\django\_projects\fbvEmployees git:[master]

py -m manage makemigrations

Migrations for 'fbvEmpApp':

fbvEmpApp\migrations\0003\_customer\_contactdetails.py

- Create model Customer

- Create model ContactDetails

~\django\_projects\fbvEmployees git:[master]

py -m manage migrate

Operations to perform:

Apply all migrations: admin, auth, contenttypes, fbvEmpApp, sessions

Running migrations:

Applying fbvEmpApp.0003\_customer\_contactdetails... OK

A screenshot of a computer

Description automatically generated

Adding records and mapping through the shell

~\django\_projects\fbvEmployees git:[master]

py -m manage shell

Python 3.12.3 (tags/v3.12.3:f6650f9, Apr 9 2024, 14:05:25) [MSC v.1938 64 bit (AMD64)] on win32

Type "help", "copyright", "credits" or "license" for more information.

(InteractiveConsole)

>>> from fbvEmpApp.models import Customer, ContactDetails

>>> customer = Customer(name='Kamal Kumar JesuRanjan')

>>> customer.save()

>>> contact = ContactDetails(type='Mobile', tvalue='0877684507', customer=customer)

>>> contact.save()

>>> contact1 = ContactDetails(type='Landline', tvalue='015039078', customer=customer)

>>> contact1.save()

>>> contact2 = ContactDetails(type='Email', tvalue='kamalrkumar@gmail.com', customer=customer)

>>> contact2.save()

>>> customer.contactdetails\_set.all()

<QuerySet [<ContactDetails: ContactDetails object (1)>, <ContactDetails: ContactDetails object (2)>, <ContactDetails: ContactDetails object (3)>]>

>>>

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

>>> customer.delete()

This will delete all the records from Customer, ContactDetails .

**One to One Relationship Table Design**

**Class:**

class Person(models.Model):  
 firstname = models.CharField(max\_length=30)  
 lastname = models.CharField(max\_length=30)  
 age = models.IntegerField(default=24)  
  
  
class License(models.Model):  
 lictype = models.CharField(max\_length=30)  
 licfor = models.CharField(max\_length=15)  
 licFrom = models.DateField(default='2024/04/01')  
 licTo = models.DateField(default='2034/04/01')  
 person = models.OneToOneField(Person, on\_delete=models.CASCADE)

**Migrate**

~\django\_projects\fbvEmployees git:[master]

py -m manage makemigrations

Migrations for 'fbvEmpApp':

fbvEmpApp\migrations\0004\_person\_license.py

- Create model Person

- Create model License

~\django\_projects\fbvEmployees git:[master]

py -m manage migrate

Operations to perform:

Apply all migrations: admin, auth, contenttypes, fbvEmpApp, sessions

Running migrations:

Applying fbvEmpApp.0004\_person\_license... OK

**Output in MySQL Workbench**

A screenshot of a computer

Description automatically generated

Command executed in Shell

py -m manage shell

Python 3.12.3 (tags/v3.12.3:f6650f9, Apr 9 2024, 14:05:25) [MSC v.1938 64 bit (AMD64)] on win32

Type "help", "copyright", "credits" or "license" for more information.

(InteractiveConsole)

>>> from fbvEmpApp.models import Person, License

>>> from datetime import date

>>> p1 = Person(firstname = 'Kamal Kumar', lastname='Jesuranjan', age=47)

>>> p1.save()

>>> p2 = Person(firstname = 'Augusta', lastname='Raju', age=47)

>>> p2.save()

>>> p3 = Person(firstname = 'Jazlyn', lastname='Kumar', age=18)

>>> p3.save()

>>> l1 = License(lictype='Permanent', licfor='Car', licFrom=date(2024,5,1),licTo=date(2029,5,31),person=p1)

>>> l1.save()

>>> l2 = License(lictype='Permanent', licfor='Car', licFrom=date(2024,7,1),licTo=date(2029,7,31),person=p2)

>>> l2.save()

>>> l3 = License(lictype='Temporary', licfor='Car', licFrom=date(2026,1,1),licTo=date(2030,12,31),person=p3)

>>> l3.save()

>>> A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Retrieve License and Person object.**

>>> p1.license

<License: License object (1)>

>>> p2.license

<License: License object (2)>

>>> p3.license

<License: License object (3)>

>>>

>>>

>>> l1.person

<Person: Person object (1)>

>>> l2.person

<Person: Person object (2)>

>>> l3.person

<Person: Person object (3)>

>>>

>>>

>>> p1.delete 🡺 Will delete both the person and license

>>> l1.person 🡺 Will only delete the licence of Person