Московский Государственный Университет имени Н. Э. Баумана

Отчет по лабораторной работе №6

по курсу разработка интернет приложений

«Работа с СУБД»

Вариант №8

Выполнила:

Студентка группы ИУ5-54

Журавлева Ульяна

Проверил:

Преподаватель кафедры ИУ5

Гапанюк Ю.У.

Москва, 2016год.

Содержание

1. Задание
2. Текст программы
3. Результаты

1. Задание

В этой лабораторной работе вы познакомитесь с популярной СУБД MySQL, создадите свою базу данных. Также вам нужно будет дополнить свои классы предметной области, связав их с созданной базой. После этого вы создадите свои модели с помощью Django ORM, отобразите объекты из БД с помощью этих моделей и ClassBasedViews.

Для сдачи вы должны иметь:

1. Скрипт с подключением к БД и несколькими запросами.

2. Набор классов вашей предметной области с привязкой к СУБД (класс должен уметь хотя бы получать нужные записи из БД и преобразовывать их в объекты этого класса)

3. Модели вашей предметной области

4. View для отображения списка ваших сущностей

2. Текст программы

Файл connection.py:

```
import MySQLdb


class Connection:
    def __init__(self, user, password, db, host='localhost'):
        self.user = user
        self.host = host
        self.password = password
        self.db = db
        self._connection = None

    @property
    def connection(self):
        return self._connection

    def __enter__(self):
        self.connect()

    def __exit__(self, exc_type, exc_val, exc_tb):
        self.disconnect()

    def connect(self):
        if not self._connection:
            self._connection = MySQLdb.connect(
                host=self.host,
                user=self.user,
                passwd=self.password,
                db=self.db,
                charset='utf8',
                use_unicode=True

            )
```

```
    def disconnect(self):
        if self._connection:
            self._connection.close()


class Group:
    def __init__(self, db_connection, group_name, base_date, genre):
        self.db_connection = db_connection.connection
        self.group_name = group_name
        self.base_date = base_date
        self.genre = genre

    def save(self):
        c = self.db_connection.cursor()
        c.execute('INSERT INTO my_app_groupmodel (group_name, base_date,
genre) VALUES (%s, %s, %s)',
                  (self.group_name, self.base_date, self.genre))
        self.db_connection.commit()
        c.close()

    def update(self):
        c = self.db_connection.cursor()
        c.execute('UPDATE my_app_groupmodel SET genre = "Рок" WHERE id=10')
        self.db_connection.commit()
        c.close()

    def delete_item(self):
        c=self.db_connection.cursor()
        c.execute('DELETE FROM my_app_groupmodel WHERE id=11')
        self.db_connection.commit()
        c.close()

con = Connection('dbuser', '123', 'music_db')

with con:
    group = Group(con, 'Альфа', '1983-01-01', 'Рок, Хард-рок, Поп-рок')
    group.save()
    group.delete_item()
    group.update()
```

Файл models.py:

```
from django.db import models


class GroupModel (models.Model):
    group_name = models.CharField(max_length=20, unique=True)   #музыкальная
группа
    base_date = models.DateField()
    genre = models.CharField(max_length=255)


class MembershipModel (models.Model): #членство
    membership_num = models.IntegerField(unique=True)
    join_date = models.DateField()
    out_date = models.DateField()
    group = models.ForeignKey('GroupModel', null=True)


class MemberModel (models.Model):
    membership_num = models.ForeignKey('MembershipModel', null=True)
    member_name = models.CharField(max_length=20)
```

```
    member_surname = models.CharField(max_length=20)
    member_thirdname = models.CharField(max_length=20)
    member_bdate = models.DateField()
```

Файл urls.py:

```
from django.conf.urls import url
from django.contrib import admin
from my_app.views import GroupView

urlpatterns = [
    url(r'^admin/', admin.site.urls),
    url(r'^group/', GroupView.as_view())
]
```

Файл views.py:

```
from django.shortcuts import render
from django.conf.urls import url
from django.views.generic import View
from my_app.models import GroupModel


class GroupView(View):
    def get(self, request):
        group = GroupModel.objects.all()
        return render(request, 'group.html', {'group':group})
```

Файл base.html:

```
{% load static %}

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel='stylesheet' type = 'text/css', href='{% static
"s/css/bootstrap.min.css" %}'>
    <title>{% block title %}{% endblock %}</title>
</head>
<body>

<div class="header">
    <div class="container" align="center">
        <h1>{% block visibletitle %}{% endblock %}</h1>
    </div>
</div>

<div>
    {% block body %}{% endblock %}
</div>
</body>
</html>
```

Файл group.html:

```
{% extends 'base.html' %}

{% block title %}{% endblock %}
```

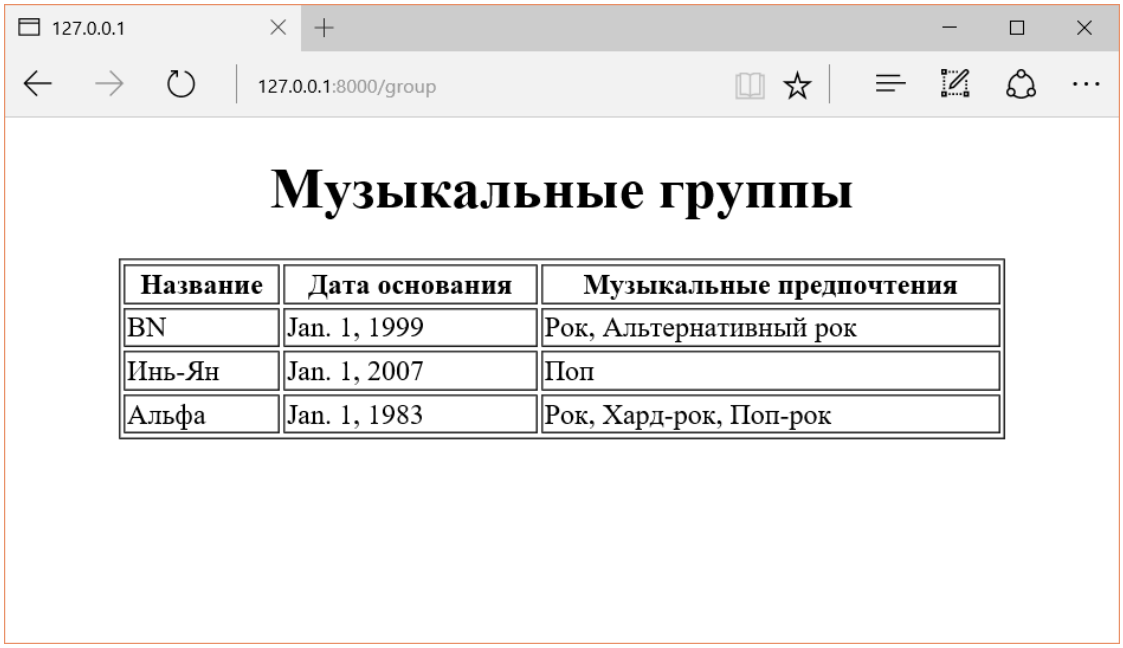
```
{% block visibletitle %}Музыкальные группы{% endblock %}

{% block body %}
    <table border="1" align="center" width="500">
        <tr><th align="middle">Название</th><th>Дата
основания</th><th>Музыкальные предпочтения
        {% for el in group %}
            <tr><td>{{ el.group_name }}</td><td>{{ el.base_date }}</td><td>{{
el.genre }}
        {% empty %}
            <tr><td colspan="3" align="center">Нет групп</td></tr>
        {% endfor %}
    </table>
{% endblock %}
```

Изменения в файле settings.py:

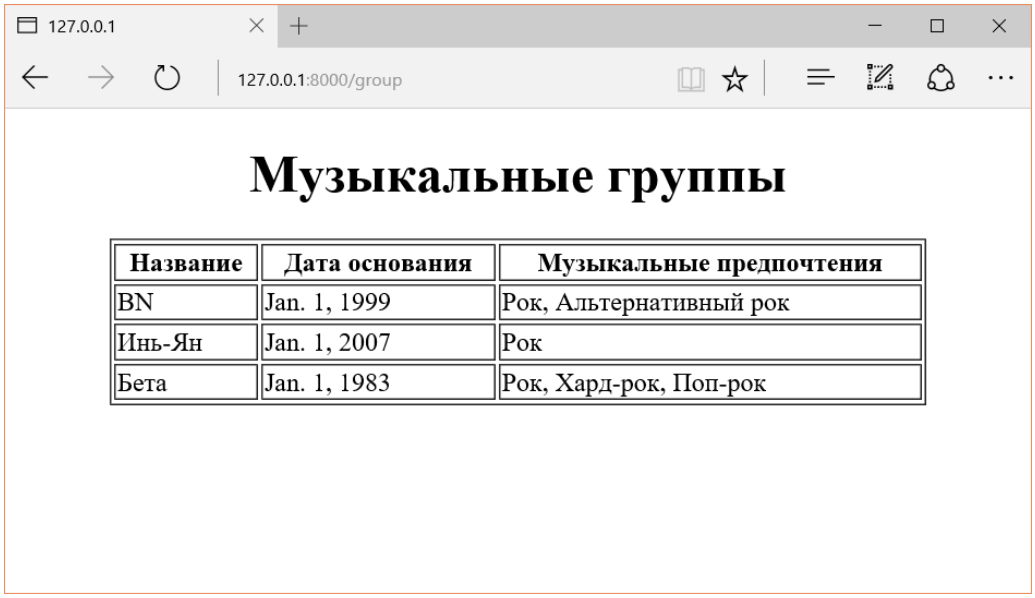
```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'music_db',
        'USER': 'dbuser',
        'PASSWORD': '123',
        'HOST': 'localhost',
        'PORT': 3306,  # Стандартный порт MySQL
        'OPTIONS': {'charset': 'utf8'},
        'TEST_CHARSET': 'utf8',
         }
    }
```

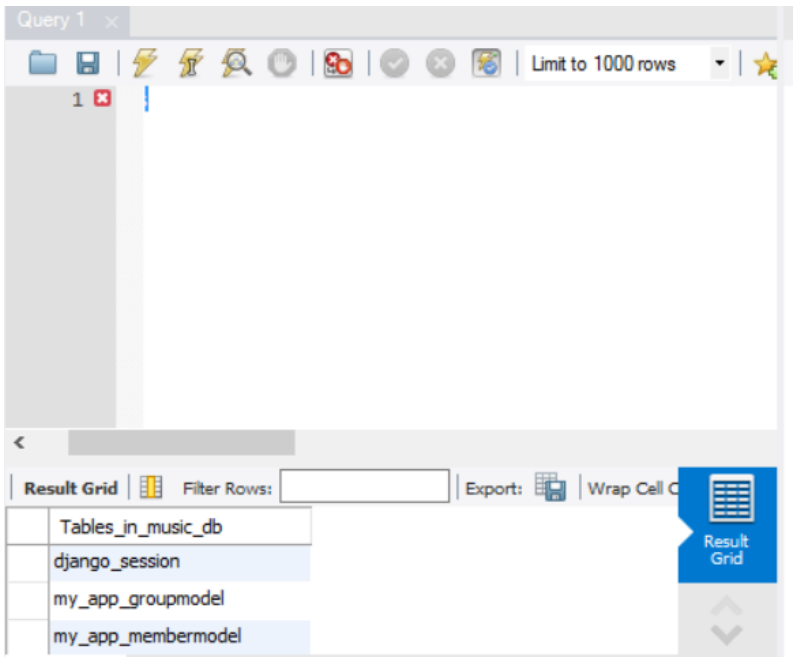
3. Результаты

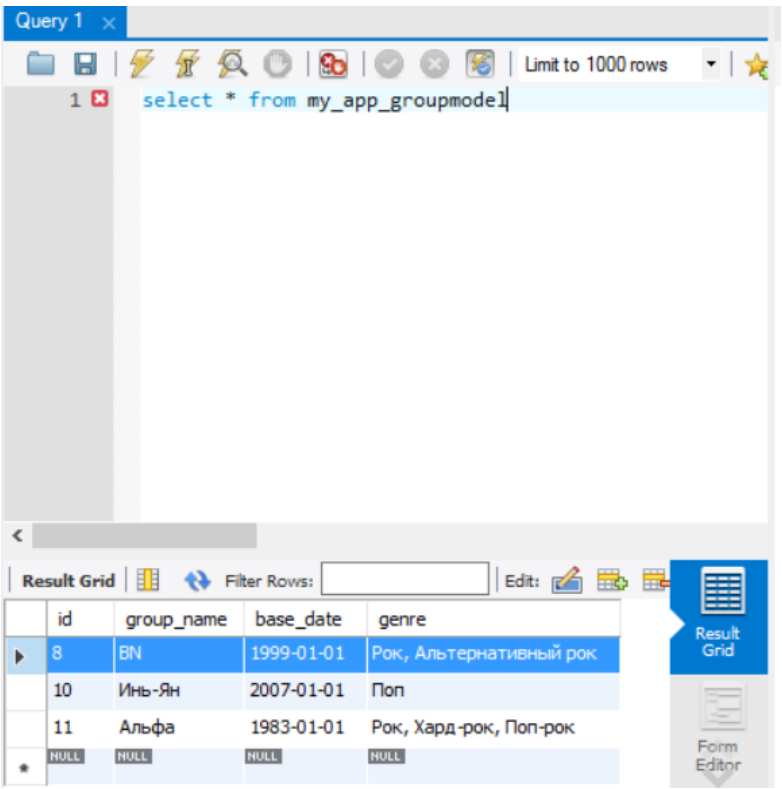
До загрузки



После загрузки


127.0.0.1
127.0.0.1:8000/group
Музыкальные группы
Название
Дата основания
Музыкальные предпочтения
BN
Jan. 1, 1999
Рок, Альтернативный рок
Инь-Ян
Jan. 1, 2007
Рок
Бета
Jan. 1, 1983
Рок, Хард-рок, Поп-рок


Query 1
Limit to 1000 rows
1
Result Grid
Filter Rows:
Export:
Wrap Cell C
Tables_in_music_db
django_session
my_app_groupmodel
my_app_membermodel
Result Grid


Query 1
Limit to 1000 rows
1
select * from my_app_groupmodel
Result Grid
Filter Rows:
Edit:
id
group_name
base_date
genre
8
BN
1999-01-01
Рок, Альтернативный рок
10
Инь-Ян
2007-01-01
Поп
11
Альфа
1983-01-01
Рок, Хард-рок, Поп-рок
NULL
Result Grid
Form Editor