**豆瓣Top250**

**代码：**

# -\*-coding:utf-8 -\*-

import io

import sys

#改变标准输出的默认编码

sys.stdout=io.TextIOWrapper(sys.stdout.buffer,encoding='utf8')

import requests

import MySQLdb

from lxml import etree

def get\_page(start\_num):

url = 'https://movie.douban.com/top250?start=%s&filter='%start\_num

res = requests.get(url)

tree = etree.HTML(res.text)

top250 = tree.xpath('//span[@class="title"][1]/text()')

print(top250)

return top250

top250 = get\_page(0)

print(top250)

def get\_all\_page(start,end):

result = []

for i in range(start,end-start):

title\_list = get\_page(i\*25)

result += title\_list

return result

topMovies = []

topMovies= get\_all\_page(0,10)

url = 'https://movie.douban.com/top250?start=%s&filter='

conn = MySQLdb.connect(host='localhost',user='root',passwd='123456',db='world',charset='utf8')

cur = conn.cursor()

for i in topMovies:

cur.execute("INSERT INTO testmodle\_test(id,content) VALUES(%s, %s)",(id,str(i)))

cur.close()

conn.commit()

conn.close()

**数据库信息：**



**天气状况爬取：**

from bs4 import BeautifulSoup

from bs4 import UnicodeDammit

import urllib.request

import sqlite3

# -\*-coding:utf-8 -\*-

import io

import sys

#改变标准输出的默认编码

sys.stdout=io.TextIOWrapper(sys.stdout.buffer,encoding='utf8')

import requests

import MySQLdb

from lxml import etree

def insert(city,date,weather,temp):

try:

conn = MySQLdb.connect(host='localhost',user='root',passwd='123456',db='world',charset='utf8')

cur = conn.cursor()

cur.execute("INSERT INTO testmodle\_weathers(City,Date,weather,Teap) VALUES(%s,%s,%s,%s)",(city,date,weather,temp))

cur.close()

conn.commit()

conn.close()

except Exception as err:

print(err)

class WeatherForecast:

def \_\_init\_\_(self):

self.headers = {"User-Agent":"Mozilla/5.0(Windows;U;Windows NT 6.0 x64;en-US;rv:1.9pre) Gecko/2008072421 Minefield/3.0.2pre"}

self.cityCode = {"北京":"101010100","南宁":"101300101","上海":"101020100","广州":"101280101"}

def forecastCity(self,city):

if city not in self.cityCode.keys():

print (city+"code cannot be found")

return

url="http://www.weather.com.cn/weather/"+self.cityCode[city]+".shtml"

try:

req = urllib.request.Request(url,headers=self.headers)

data = urllib.request.urlopen(req)

data = data.read()

dammit = UnicodeDammit(data,["utf-8","gbk"])

data = dammit.unicode\_markup

soup = BeautifulSoup(data,"lxml")

lis = soup.select("ul[class='t clearfix'] li")

n=0

for li in lis:

try:

date = li.select('h1')[0].text

weather = li.select('p[class="wea"]')[0].text

if n>0:

temp = li.select('p[class="tem"] span')[0].text + "/" + li.select('p[class="tem"] i')[0].text

else:

temp = li.select('p[class="tem"] i')[0].text

print(city,date,weather,temp)

insert(city,date,weather,temp)

n=n+1

except Exception as err:

print(err)

except Exception as err:

print(err)

def process(self,cities):

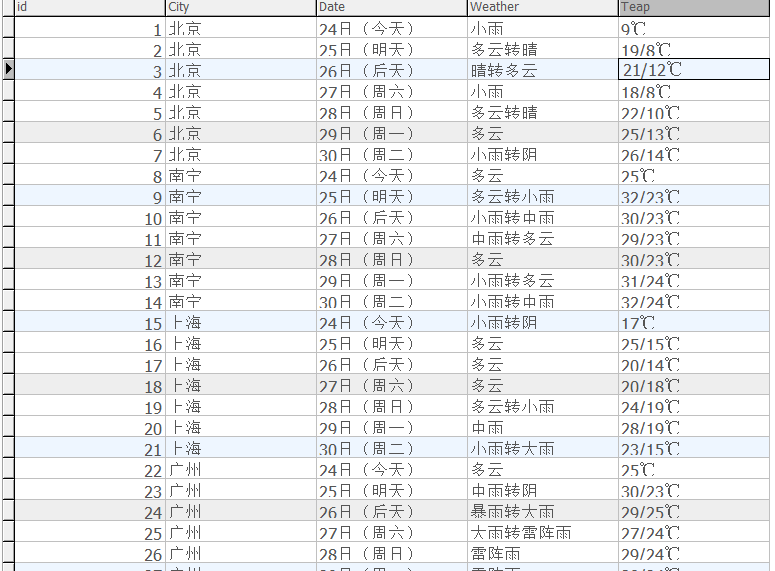
for city in cities:

self.forecastCity(city)

ws = WeatherForecast()

ws.process(["北京","南宁","上海","广州"])

**数据库信息：**



**使用Selenium爬取动态网页：**

#coding:utf-8

from selenium import webdriver

import os

import time

import pymysql.cursors

class Crawler(object):

def \_\_init\_\_(self):

self.Chromedriver = "C:/Program Files (x86)/Google/Chrome/Application/chromedriver.exe"

os.environ["webdriver.chrome.driver"] = self.Chromedriver

self.Chrome = webdriver.Chrome(self.Chromedriver)

# Connect to the database

self.MySql = pymysql.connect(host='localhost',

user='root',

password='123456',

db='world',

charset='utf8',

cursorclass=pymysql.cursors.DictCursor)

def crawling36kr(self):

url = "https://www.36kr.com/information/web\_news"

# 添加cookie前必须先打开一次网站

self.Chrome.get(url)

cookie = { "name" : "new\_user\_guidance", "value" : "true", "domain" : ".36kr.com"}

self.Chrome.add\_cookie(cookie)

self.Chrome.get(url)

time.sleep(3)

item\_list = self.Chrome.find\_elements\_by\_class\_name("kr-shadow-content")

print("item\_list = ", len(item\_list))

index = 0

while index < len(item\_list) - 1:

item = item\_list[index]

index += 1

item.location\_once\_scrolled\_into\_view

imgSrc = item.find\_element\_by\_class\_name("scaleBig").get\_attribute("src")

title = item.find\_element\_by\_class\_name("article-item-title")

path = title.get\_attribute("href")

print(index, " len = ", len(item\_list), " img = " , imgSrc , " path = " , path, " title = ", title.text)

self.news\_detail\_36kr(index, item)

item\_list = self.Chrome.find\_elements\_by\_class\_name("kr-shadow-content")

time.sleep(2)

if index == len(item\_list) - 1:

loading\_more\_button = self.Chrome.find\_element\_by\_class\_name("kr-loading-more-button")

loading\_more\_button.location\_once\_scrolled\_into\_view

time.sleep(3)

item\_list = self.Chrome.find\_elements\_by\_class\_name("kr-shadow-content")

if index == len(item\_list) - 1:

loading\_more\_button.click()

time.sleep(3)

item\_list = self.Chrome.find\_elements\_by\_class\_name("kr-shadow-content")

def news\_detail\_36kr(self, index, item):

imgSrc = item.find\_element\_by\_class\_name("scaleBig").get\_attribute("src")

title = item.find\_element\_by\_class\_name("article-item-title")

path = title.get\_attribute("href")

title.click()

# select second page

num = self.Chrome.window\_handles

self.Chrome.switch\_to\_window(num[1])

time.sleep(5)

# get content

article\_title = self.Chrome.find\_element\_by\_class\_name("article-title")

author = self.Chrome.find\_element\_by\_class\_name("title-icon-item")

summary = self.Chrome.find\_element\_by\_class\_name("summary")

p\_list = self.Chrome.find\_elements\_by\_xpath("//\*[@id='app']/div/div/div/div/div/div/div/div/div/div/div/div/div/div/div/p")

print("article\_title == ", article\_title.text, " author = ", author.text, " summary = ", summary.text, " len(p\_list) = ", len(p\_list))

content = ""

for p in p\_list:

content += p.text

print(content)

# insert Data to db

# try:

try:

with self.MySql.cursor() as cursor:

# Create a new record

sql = "INSERT INTO `testmodle\_dontai` (`title`, `url`, `img`, `author`, `summary`, `content`) VALUES (%s, %s, %s, %s, %s, %s)"

cursor.execute(sql, (article\_title.text, path, imgSrc, author.text, summary.text, content))

# connection is not autocommit by default. So you must commit to save your changes.

self.MySql.commit()

except Exception as e:

print(e)

time.sleep(3)

self.Chrome.close()

self.Chrome.switch\_to\_window(num[0])

if \_\_name\_\_ == "\_\_main\_\_":

crawler = Crawler()

crawler.crawling36kr()

**数据库信息：**



**五、实验结果与分析（**含程序、数据记录及分析和实验总结等**）：**

**Django的使用**

**Settings.py：**

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

'TestModle',

]

T

BACKEND': 'django.template.backends.django.DjangoTemplates',

'DIRS': [BASE\_DIR+"/templates",],

'APP\_DIRS': True,

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.mysql',

'NAME': 'world',

'USER': 'root',

'PASSWORD':'123456',

'HOST':'localhost',

'PORT':'3306',

}

**Testdb.py：**

from django.http import HttpResponse

from TestModle.models import Test

from TestModle.models import weathers

from TestModle.models import dontai

from django.shortcuts import render

import MySQLdb

#数据库操作

def testdb(request):

allList = Test.objects.all()#获取top250电影

weather = weathers.objects.all()#获取天

dontais=dontai.objects.all()#获取动态信息

Test.objects.order\_by("id")

return render(request,'base1.html',{'allList':allList,'weather':weather,'dontais':dontais})

**Urls.py：**

from django.urls import path,include

from . import view,testdb

from TestModle import views

urlpatterns = [

path(r'testdb/', testdb.testdb),

]

**Views.py**

from django.http import HttpResponse

from django.shortcuts import render

import MySQLdb

**templates/html：**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

</head>

<body>

<div class="jumbotron text-center" style="margin-bottom:0">

<h1>Django显示页面</h1>

</div>

<nav class="navbar navbar-inverse">

<div class="container-fluid">

<div class="navbar-header">

<button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#myNavbar">

<span class="icon-bar"></span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

</button>

<a class="navbar-brand" href="#">网站名</a>

</div>

</div>

</nav>

<div class="container">

<div class="row">

<div class="col-sm-4">

<h2>席子文</h2>

<h5></h5>

<h3></h3>

<p></p>

<ul class="nav nav-pills nav-stacked">

<li class="active"><a href="https://github.com/16219111431/Django" target="\_blank">github</a></li>

</ul>

<hr class="hidden-sm hidden-md hidden-lg">

</div>

<h2>天气爬取</h2>

<div class="panel-group" id="accordion">

<div class="panel panel-default">

<div class="panel-heading">

<h4 class="panel-title">

<a data-toggle="collapse" data-parent="#accordion"

href="#collapseOne">

七日天气预报

</a>

</h4>

</div>

<div id="collapseOne" class="panel-collapse collapse">

<div class="panel-body">

{% for i in weather %}

<li style="font-size:large">{{ i.City }} {{i.Date}} {{i.Weather}} {{i.Teap}}</li>

<hr />

{% endfor %}

</div>

</div>

</div>

<br>

<h2>豆瓣Top250电影</h2>

<div class="fakeimg">图像</div>

<div class="container"></div>

<pre class="pre-scrollable"><!--内容可滚动-->

<ol >

{% for i in allList %}

<li>{{ i.content }}</li>

{% endfor %}

</ol>

</pre>

</div>

<h2>动态信息</h2>

<div class="dt"></div>

<pre class="pre-scrollable">

<ol>

{% for i in dontais %}

<li>{{i.title}} {{i.url}} {{i.img}} {{i.author}} {{i.summary}} {{i.content}} </li>

{% endfor %}

</ol>

</pre>

</body>

</html>

**TestModle/Models.py：**

from django.db import models

from django.db import models

class Test(models.Model):

content = models.CharField(max\_length=255,default ="")

objects = models.Manager()

class weathers(models.Model):

City=models.CharField(max\_length=20)

Date =models.CharField(max\_length=20)

Weather=models.CharField(max\_length=80)

Teap=models.CharField(max\_length=40)

objects=models.Manager

class dontai(models.Model):

title=models.CharField(max\_length=50)

url=models.CharField(max\_length=80)

img=models.CharField(max\_length=80)

author=models.CharField(max\_length=80)

summary=models.CharField(max\_length=80)

content=models.CharField(max\_length=80)

**页面显示：**

