**基于Python的当当网商品信息爬虫系统说明文档**

**1 系统运行环境**

本系统为简单的爬虫系统，开发环境为Windows 10、Python 3.7和MySQL 8.0.18，故使用本系统需安装：

1.1 服务器

服务器版本：8.0.18

主机：localhost

端口：3306

用户名：root

密码：cxks20121221！

数据库名称：dangdang

连接编码：utf8mb4

1.2 数据库

系统中爬取到的数据存储用到了数据库，数据库版本号为MySQL 8.0.18。

1.3 浏览器

只要能运行浏览器的计算机，并支持的浏览器就可以正确执行。此系统使用360安全浏览器调试制作。

**2 系统功能分析**

2.1 爬取功能分析

（1）所需爬取的数据

以当当网（dangdang.com）为例，所需要爬取的数据为商品标题、商品出售店铺、商品链接、商品评论数、商品价格这些直观信息。

（2）用python获取当当网的源代码

爬虫最主要的处理对象就是URL，它根据URL地址取得所需要的文件内容，然后对它进行进一步的处理。因此，准确地理解URL对理解网络爬虫至关重要。

URL是URI的一个子集。它是Uniform Resource Locator的缩写，译为“统一资源定位符”。通俗地说，URL是Internet上描述信息资源的字符串，主要用在各种WWW客户程序和服务器程序上。采用URL可以用一种统一的格式来描述各种信息资源，包括文件、服务器的地址和目录等。URL的格式由三部分组成： ①第一部分是协议(或称为服务方式)。②第二部分是存有该资源的主机IP地址(有时也包括端口号)。③第三部分是主机资源的具体地址，如目录和文件名等。其中，第一部分和第二部分用“://”符号隔开，第二部分和第三部分用“/”符号隔开。第一部分和第二部分是不可缺少的，第三部分有时可以省略。

Python获取网页源代码可用urllib或urllib2函数进行，极其方便快捷，代码如下：

import urllib2

response = urllib2.urlopen('http://www.baidu.com/')

html = response.read()

print html

（3）应用python伪装成浏览器自动登陆当当网，加入变量打开多个网页。

有了源代码就可以进行数据的爬取了，在获取数据之前，需要对爬虫进行一下伪装，使其成为一个浏览器，以实现全网页的爬取工作。

应用opener和header的基础知识即可实现伪装成浏览器这一步骤。在伪装的同时，需要加入变量来打开多个网页，是的爬取工作可以顺利进行。

（4）用正则表达式分析网页源代码

正则表达式是用于处理字符串的强大工具，它并不是Python的一部分。其他编程语言中也有正则表达式的概念，区别只在于不同的编程语言实现支持的语法数量不同。它拥有自己独特的语法以及一个独立的处理引擎，在提供了正则表达式的语言里，正则表达式的语法都是一样的。

正则表达式的大致匹配过程是：依次拿出表达式和文本中的字符比较，如果每一个字符都能匹配，则匹配成功；一旦有匹配不成功的字符则匹配失败。如果表达式中有量词或边界，这个过程会稍微有一些不同。

Python是通过re模块实现对正则表达式的调用的。程序代码如下：

find\_re = re.compile(r"absolute.+?span>(.\*?)</.+?&gt;&gt;</a>(.\*?)</h2>.+?</b>(

.\*?)</s.+?</b>(.\*?)</s.+?</b>(.\*?)</sp.+?</b>(.\*?)</sp.+?</b>", re.DOTALL)

2.2 存储功能分析

数据储存在数据库中。数据库指的是以一定方式储存在一起、能为多个用户共享、具有尽可能小的冗余度、与应用程序彼此独立的数据集合。将数据储存在数据库中也具有直观简洁的特点。

Python调用数据库是用pymysql模块。

3 **相关技术说明**

3.1 爬虫的介绍

网络爬虫按照系统结构和实现技术，大致可以分为以下几种类型：通用网络爬虫（General Purpose Web Crawler）、聚焦网络爬虫（Focused Web Crawler）、增量式网络爬虫（Incremental Web Crawler）、深层网络爬虫（Deep Web Crawler）。实际的网络爬虫系统通常是几种爬虫技术相结合实现的。

通用网络爬虫是捜索引擎抓取系统（Baidu、 Google、 Yahoo 等）的重要组成部分。 主要目的是将互联网上的网页下载到本地，形成一个互联网内容的镜像备份。

通用网络爬虫从互联网中搜集网页，采集信息，这些网页信息用于为搜索引擎建立索引从而提供支持，它决定着整个引擎系统的内容是否丰富，信息是否即时，因此其性能的优劣直接影响着搜索引擎的效果。

搜索引擎网络爬虫的基本工作流程：

（1）首先选取一部分的种子URL，将这些URL放入待抓取 URL 队列；

（2）取出待抓取URL解析DNS得到主机的IP，并将URL对应的网页下载下来，存储进已下载网页库中，并且将这些URL放进已抓取 URL 队列。

（3）分析已抓取URL队列中的URL，分析其中的其他URL，并且将URL放入待抓取URL队列，从而进入下一个循环。

搜索引擎获取一个新网站的URL的方法:

（1）新网站向搜索引擎主动提交网址

（2）在其他网站上设置新网站外链（尽可能处于搜索引擎爬虫爬取范围）

（3）搜索引擎和DNS解析服务商(如DNSPod等）合作，新网站域名将被迅速抓取。

但是搜索引擎蜘蛛的爬行是被输入了一定的规则的，它需要遵从一些命令或文件的内

容，如标注为no follow的链接，或者是Robots协议。

Robots协议（也叫爬虫协议、机器人协议等），全称是“网络爬虫排除标准”（Robots Exclusion Protocol），网站通过Robots协议告诉搜索引擎哪些页面可以抓取，哪些页面不能抓取。

3.2 Python爬虫架构

Python 爬虫架构主要由五个部分组成，分别是调度器、URL管理器、网页下载器、网页解析器、应用程序（爬取的有价值数据）。

（1）调度器：相当于一台电脑的CPU，主要负责调度URL管理器、下载器、解析器之间的协调工作。

（2）URL管理器：包括待爬取的URL地址和已爬取的URL地址，防止重复抓取URL和循环抓取URL，实现URL管理器主要用三种方式，通过内存、数据库、缓存数据库来实现。

（3）网页下载器：通过传入一个URL地址来下载网页，将网页转换成一个字符串，网页下载器有urllib2（Python官方基础模块）包括需要登录、代理、和cookie，requests(第三方包)

（4）网页解析器：将一个网页字符串进行解析，可以按照我们的要求来提取出我们有用的信息，也可以根据DOM树的解析方式来解析。网页解析器有正则表达式（直观，将网页转成字符串通过模糊匹配的方式来提取有价值的信息，当文档比较复杂的时候，该方法提取数据的时候就会非常的困难）、html.parser（Python自带的）、beautifulsoup（第三方插件，可以使用Python自带的html.parser进行解析，也可以使用lxml进行解析，相对于其他几种来说要强大一些）、lxml（第三方插件，可以解析 xml 和 HTML），html.parser 和 beautifulsoup 以及 lxml 都是以 DOM 树的方式进行解析的。

（5）应用程序：就是从网页中提取的有用数据组成的一个应用。

3.3 第三方库 Beautiful Soup 的安装

Beautiful Soup是Python的第三方插件，用来提取xml和HTML 中的数据，官网地址为： [https://www.crummy.com/software/BeautifulSoup/](https://www.crummy.com/software/BeautifulSoup/" \t "https://www.runoob.com/w3cnote/_blank)

1、安装 Beautiful Soup

打开 cmd（命令提示符），进入到 Python（Python2.7版本）安装目录中的 scripts 下，输入 dir 查看是否有 pip.exe, 如果用就可以使用 Python 自带的 pip 命令进行安装，输入以下命令进行安装即可：

pip install beautifulsoup4

2、测试是否安装成功

编写一个 Python 文件，输入:

import bs4print bs4

运行该文件，如果能够正常输出则安装成功。

3.4 使用Beautiful Soup解析html文件

#!/usr/bin/python

# -\*- coding: UTF-8 -\*-

import re

from bs4 import BeautifulSoup

html\_doc = """

<html><head><title>The Dormouse's story</title></head>

<body>

<p class="title"><b>The Dormouse's story</b></p>

<p class="story">Once upon a time there were three little sisters; and their names were

<a href="http://example.com/elsie" class="sister" id="link1">Elsie</a>,

<a href="http://example.com/lacie" class="sister" id="link2">Lacie</a> and

<a href="http://example.com/tillie" class="sister" id="link3">Tillie</a>

; and they lived at the bottom of a well.</p>

<p class="story">...</p>

"""

#创建一个BeautifulSoup解析对象

soup = BeautifulSoup(html\_doc,"html.parser",from\_encoding="utf-8")

#获取所有的链接

links = soup.find\_all('a')

print "所有的链接"

for link in links:

print link.name,link['href'],link.get\_text()

print "获取特定的URL地址"

link\_node = soup.find('a',href="http://example.com/elsie")

print link\_node.name,link\_node['href'],link\_node['class'],link\_node.get\_text()

print "正则表达式匹配"

link\_node = soup.find('a',href=re.compile(r"ti"))

print link\_node.name,link\_node['href'],link\_node['class'],link\_node.get\_text()

print "获取P段落的文字"

p\_node = soup.find('p',class\_='story')

print p\_node.name,p\_node['class'],p\_node.get\_text()

3.5 pymysql.connect()参数说明

（1）pymysql.Connect()参数说明

host(str MySQL服务器地址

port(int) MySQL服务器端口号

user(str) 用户名

passwd(str) 密码

db(str) 数据库名称

charset(str) 连接编码

（2）connection对象支持的方法

cursor() 使用该连接创建并返回游标

commit() 提交当前事务

rollback() 回滚当前事务

close() 关闭连接

（3）cursor对象支持的方法

execute(op) 执行一个数据库的查询命令

fetchone() 取得结果集的下一行

fetchmany(size) 获取结果集的下几行

fetchall() 获取结果集中的所有行

rowcount() 返回数据条数或影响行数

close() 关闭游标对象

3.6 scrapy中使用cmdline快捷运行项目

在scrapy中，为了避免每一次运行或调试都输入一串命令，可以在项目文件下新建一个run.py文件，每次运行爬虫只需要运行此脚本即可。且运行调试模式也需要设置此启动脚本。

from scrapy import cmdline

cmdline.execute('scrapy crawl zhihu\_topic'.split())

若嫌弃scrapy日志文件太杂乱,想无日志输出,只需在后面增加--nolog即可:

from scrapy import cmdline

cmdline.execute('scrapy crawl zhihu\_topic --nolog'.split())

以上代码只需要将我的''zhihu\_topic''替换为你自己的爬虫名就可以了。

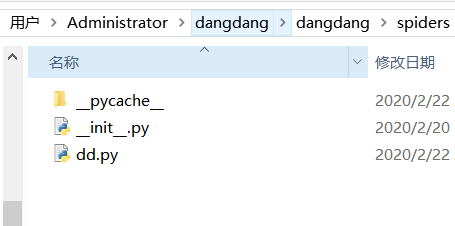
**4 主要设计和实现**

4.1 创建爬虫项目

具体操作：打开cmd界面，用cd操作跳转到你想要存储文件的位置，输入：scrapy startproject dangdang，创建dangdang爬虫项目

4.2 创建一个以基础爬虫为模板的爬虫文件，命令为：dd.py

具体操作：在cmd界面输入：scrapy genspider -t dd dangdang.com即可。结果如下图所示，在spiders文件夹中生成了文件dd.py。



4.3 确定需要爬取的数据

具体操作：在配置文件items.py中创建要爬取的数据定义，具体代码如下：

import scrapy

class DangdangItem(scrapy.Item):

# define the fields for your item here like:

# name = scrapy.Field()

title = scrapy.Field()

link = scrapy.Field()

comment = scrapy.Field()

shop = scrapy.Field()

price = scrapy.Field()

4.4 确定网页以及数据的命名规律

具体操作：打开当当网（dangdang.com）,按F12查看网页源代码，确定想要爬虫的数据在网页中的命名方式。同时找寻多个网页地址的规律，便于编程。在爬虫文件dd.py中基于Xpath爬取网站信息，并且实现多页爬取。具体代码如下：

import sys

import os

curPath = os.path.abspath(os.path.dirname(\_\_file\_\_))

rootPath = os.path.split(curPath)[0]

sys.path.append(rootPath)

import scrapy

from dangdang.items import DangdangItem

from scrapy.http import Request # 依次爬取的工具

class DdSpider(scrapy.Spider):

name = 'dd'

allowed\_domains = ['dangdang.com']

start\_urls = ['http://category.dangdang.com/pgl-cid4008154.html']

header = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64:x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3282.140 Safari/537.36',}

def parse(self, response):

item = DangdangItem()

item["title"] = response.xpath('//p[@name="title"]/a/@title').extract()

item["link"] = response.xpath('//p[@name="title"]/a/@href').extract()

item["comment"] = response.xpath('//a[@dd\_name="单品评论"]/text()').extract()

item["shop"] = response.xpath('//p/a[@dd\_name="单品店铺"]/text()').extract()

item["price"] = response.xpath('//span[@class="price\_n"]/text()').extract()

print(item)

yield item # 将数据提交到Pipeline中

for i in range(2,2001):

url = 'http://category.dangdang.com/pg'+str(i)+'-cid4008154.html'

# callback为回调函数

yield Request(url,callback=self.parse,headers=self.header)

4.5 爬虫风险规避

由于可能会出现爬虫失败，以及我们需要将获取的数据进行相关处理，因此我们在settings.py文件处，要修改三处：

① BOT\_NAME = 'dangdang

SPIDER\_MODULES = ['dangdang.spiders']

NEWSPIDER\_MODULE = 'dangdang.spiders'

② ROBOTSTXT\_OBEY = False

③ ITEM\_PIPELINES = { 'dangdang.pipelines.DangdangPipeline': 300,}

4.6 创建数据库和表

本系统选择的是MySQL数据库，采用Navicat for MySQL实现系统与数据库的简单连接。打开Navicat ，新建数据库，并且创建相应的SQL表，用于存储数据，命名为products。查询代码如下：

CREATE TABLE `products`(

`id` int(10) NOT NULL AUTO\_INCREMENT,

`shop` varchar(100) DEFAULT NULL COMMENT '出售店铺',

`title` varchar(100) DEFAULT NULL COMMENT '商品标题',

`link` varchar(100) DEFAULT NULL COMMENT '商品链接',

`price` varchar(100) DEFAULT NULL COMMENT '商品价格',

`comment` varchar(100) DEFAULT NULL COMMENT '商品评论',

PRIMARY KEY (`id`)

)ENGINE=INNODB DEFAULT CHARSET=utf8;

4.7 实现连接数据库功能

在Pipelines.py文件中编写相应的数据库链接函数，以及数据库存储操作将数据存入本地数据库中，具体代码如下：

import pymysql

def condb():

conn = pymysql.connect(host='localhost',port=3306,user='root',

password='cxks20121221!',db='dangdang',charset='utf8mb4')

return conn

class DangdangPipeline(object):

def process\_item(self, item, spider):

dbObject = condb()

cursor = dbObject.cursor()

sql = 'INSERT INTO products(shop,title,link,price,comment) VALUES(%s,%s,

%s,%s,%s)'

# 将数据导入到本地数据库中

for j in range(len(item['title'])):

try:

cursor.execute(sql,(item['shop'][j],item['title'][j],item['link'][j],

item['price'][j],item['comment'][j],))

cursor.close()

dbObject.commit()

except Exception as e:

print(e)

dbObject.rollback()

return item

4.8 运行系统功能

新建run.py，用于直接运行爬虫。具体代码如下：

from scrapy import cmdline

cmdline.execute('scrapy crawl zhihu\_topic'.split())

**5 使用方法**

直接打开run.py文件，按“Ctrl+B”运行即可。

**6 补充说明**

（1）本系统在不同电脑上使用时，只需将piplines.py中数据库服务器地址host、服务器端口号port、用户名user、密码password、数据库名称db、连接编码charset修改成与本机电脑中服务器一致，确保服务器能够连接成功，即可正确运行本程序。

（2）可以通过修改dangdang文件夹中的dd.py中i的取值范围，来改变爬虫到的数据条数。

（3）如果不希望出现scrapy日志文件，想要无日志输出，只需将语句：cmdline.execute('scrapy crawl dd'.split())修改为：cmdline.execute('scrapy crawl dd --nolog'.split())即可。