华北电力大学

课程设计报告

（ 2019--2020 年度第二学期）

名　　称：Python 课程设计

题　　目：scrapy 框架下爬取 51job 网站

院　　系：控制与计算机工程学院

班　　级：软件 1801

学　　号：120181080223

学生姓名：姚鹏飞

指导教师：熊建国

设计周数：1 周

成　　绩：

日期： 2020 年 6 月 25 日

一、课程设计(综合实验)的目的与要求

通过课程设计的实践训练，加强学生对 Python 语言基础知识、不同模块的理解运用、利用模块进行程序设计并且达成目标，提高学生综合运用所学知识分析问题、解决问题的能力。

具体如下：

1．根据题目要求进行需求分析，确定系统的功能需求、数据需求、性能需求等；

2．根据分析的结果进行软件设计，包括类的设计、数据设计等；

3．采用 Python 语言实现系统的主要功能；

4．撰写课程设计报告，要求内容完整、格式规范。

二、设计（实验）正文

1．需求分析

1.1 功能需求

目标要实现的功能包括：

（1）利用 scrapy 框架从 http://www.51job.com 网站爬取关于 Python 开发工程师的工作岗位信息，并按照一定结构保存到数据库中。

（2）根据爬取的工作信息计算北京地区的 Python 开发工程师的平均薪资。

额外实现功能有：

(1) 可以查找除了指定工作以外任意工作的工作岗位信息。

(2) 可以计算任何地区的平均薪资。

1.2 数据需求

系统需要获取并保存的数据包括：

（1）从 http://www.51job.com 网站爬取关于 Python 开发工程师的工作岗位信息

2．程序总体设计

2.1 数据库的设计

工作岗位数据表：

（1）数据库名：job_search

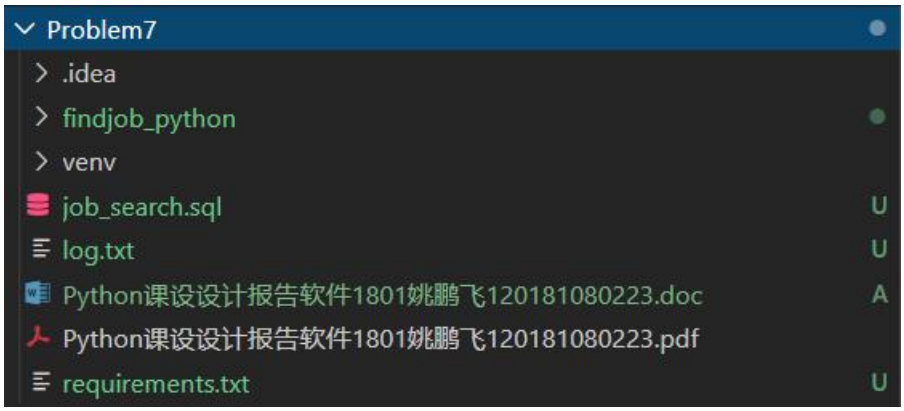
（2）表名：pythondeveloperjob

数据库表结构说明：

字段名称	字段类型	字段约束	字段说明
id	INT	主键、自增	用户唯一 id
JobTitle	VARCHAR(255)	无	工作名称
CompanyName	VARCHAR(255)	无	公司名称
WorkPosition	VARCHAR(255)	无	工作地点
Salary	FLOAT(20)	无	薪资水平
PublishTime	VARCHAR(255)	无	发布时间

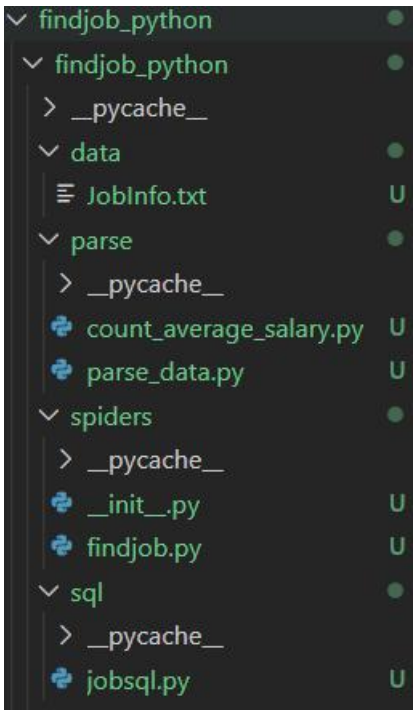
2.2 程序结构

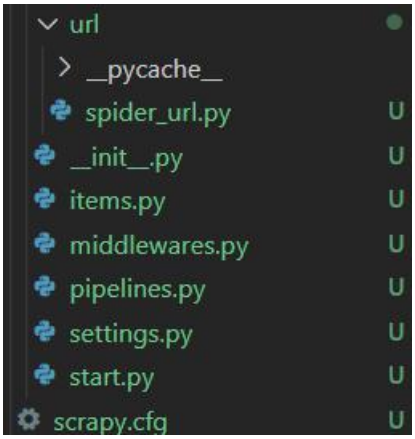
程序位置存放于 exe11/Problem7 中。一级目录结构如下：



文件名	文件类型	说明
.idea	文件夹	存放 Pycharm 配置信息
findjob_python	文件夹	存放爬虫主体程序
venv	文件夹	virtualenv 生成的虚拟环境
job_search.sql	sql 脚本	在数据库中运行该脚本即可生成所需数据库的表
log.txt	txt 文本	利用 scrapy 生成的日志文件
报告.doc	word 文档	报告的 doc 版本
报告.pdf	pdf 文件	报告的 pdf 版本
requirements.txt	txt 文本	在虚拟环境中用 pip 运行该文件进行环境配置

其中，findjob_python 文件夹下有程序主体结构，下图为文件夹内主要结构，附表格说明。





文件名	文件类型	说明
data	文件夹	存放爬取数据。
data/JobInfo.txt	txt 文本	爬取数据的 txt 文本形式，用于计算薪资。
parse	文件夹	存放解析数据的程序。
parse/count_average_salary.py	PY 文件	根据解析好的数据计算北京地区的薪资水平。
parse/parse_data.py	PY 文件	用于对网页中解析出来的文本薪资处理。
spiders	文件夹	存放 scrapy 框架中的 spiders 层的程序
spiders/__init__.py	PY 文件	spiders 层的初始化的程序
spiders/findjob.py	PY 文件	爬取 51job 网站的工作信息的 spiders 层的程序
sql	文件夹	存放管理爬虫数据的数据库相关程序
sql/jobsql.py	PY 文件	用于数据库与爬虫程序的连接、管理的程序
url	文件夹	用于存放目标访问网站的静态网址部分
url/spider_url.py	PY 文件	存放 51job.com 网站静态网址的程序
__init__.py	PY 文件	scrapy 框架初始化的程序
items.py	PY 文件	scrapy 框架的 item 结构处理程序，用于处理 engine 返回的结构化数据
middlewares.py	PY 文件	scrapy 框架的中间件处理程序
pipelines.py	PY 文件	scrapy 框架的 Pipeline 管道处理程序，用于处理结构化数据

		的存储
settings.py	PY 文件	scrapy 框架的配置文件
start.py	PY 文件	scrapy 爬虫的启动程序，封装了原本需要用户自己在终端输入的命令。

3. 程序实现

3.1 开发技术

（1）scrapy 框架：

由于爬取的网页数量在 10^3 次数量级以上，之前所学的 requests 模块的爬取速度响应时间的缺点被放大，不太能够满足爬取量大的要求。因此选择了 scrapy 框架。scrapy 虽然在灵活程度，破解反爬取方面效果不好，但是此次目标网站 51job.com 的反爬取措施不强。仅有拦截访问频率过高导致的恶意攻击 IP 的措施。所以只要一定程度上控制下载速度，scrapy 框架是一个很好的选择。除此以外，scrapy 框架的 middlewares 层采用的是 twisted 的异步结构，满足题目的多线程或多进程的要求。

（2）BeautifulSoup：

利用 scrapy 得到网页的 response 后，相当于得到了网页的 html 源码。分析网页结构之后发现使用 BeautifulSoup 比使用 Scrapy 自带的 css 选择器效果更好。故使用 bs4 来解析网页快速得到目标字段的数据。

（3）pymysql：

由于解析数据中包含大量的中文以及数字，利用 txt 文本存储得到的数据效果非常一般。故选择采用数据库来存储数据。由于在这个程序中，数据库仅用作储存数据，并不需要其他太多的功能，因此数据库首要以轻便为主不要有太多的文件结构。pymysql 相较于 SQLAchemy 而言，执行效率更高，且没有过多的配置结构，故选择 pymysql。

3.2 程序执行流程

本程序基本执行顺序基于 scrapy 框架本身的执行流程。使用时，进入虚拟环境之后在终端执行 start.py 文件。start.py 会使用 cmdline 执行 scrapy 的终端指令，启动 findjob.py。在 spiders 层中，拼接静态网址得到查询目标的网页，发出 Request，提交数据申请给 Downloadr 下载数据，并且返回生成器并且 callback 回调 parse_page 解析服务器返回的 response。parse_page 在得到 Downloader 的文件后，解析网页得到结构化数据的字典，返回一个工作信息的字典。在经过 engine 层后由 items 提取字典到 Pipeline 层，做最后的保存工作。在 Pipeline 层中，当 spider 启动时，打开 data 中的 JobInfo.txt 和数据库的连接。每当有 items 的数据传来，就按照格式分别保存。在 spiders 不再发送数据时，保存完所有数据之后 Pipelines 就结束工作。整个过程当中，Spiders，Downloader，ItemPipeline 利用 middlewares 连接，并发执行。

3.3 核心问题

（1）目标网页网址获取：

```
https://search.51job.com/list/000000,000000,0000,00,9,99,Python%25E5%25BC%2580%25E5%258F%2591%25E5%25B7%25A5%25E7%25A8%258B%25E5%25B8%2588,2,1.html?lang=c&postchannel=0000&workyear=99&cotype=99&degreefrom=99&jobterm=99&companysize=99&ord_field=0&dibiaoid=0&line=&welfare=
```

上面所展示的是 Python 开发工程师（+全国）的查询网址。经过分析比对之后发现，该代码是由三部分组成。

第一部分是静态网址，为‘https://search.51job.com/list/000000,000000,0000,00,9,99,’和‘.html?lang=c&postchannel=0000&workyear=99&cotype=99&degreefrom=99&jobte

rm=99&companysize=99&ord_field=0&dibiaoid=0&line=&welfare=’组成。这两块网址在其他检索和页码中都不会发生改变。

第二部分是查询内容，为‘Python%25E5%25BC%2580%25E5%258F%2591%25E5%25B7%25A5%25E7%25A8%258B%25E5%25B8%2588’。此段网址是将查询内容通过 url 编码之后加上 51job.com 自己使用的一个编码方式编译的。主要为：在生成的 quote 字段的每一个‘%’后加上 25。在实际使用时，即便不经过 51job.com 编译，直接采用 url 编码也可以正确访问。

第三部分是查询页码。是剩下的‘,2,1’。其中，后面一个数字代表了当前页码。当该数字超出了实际网页中的页码时，查询结果会为空，不会报错。根据多天观察，发现 Python 开发工程师的实际工作招聘信息不会超过 1000 页，因此可以设置一个常数来控制访问页面的数量。

（2）html 源码解析：

根据在 Chrome 中的开发者工具找到的 html 源码分析。解决编码问题之后可以发现目标数据均保存在 class 为‘dw_table’，id 为‘resultList’的 div 标签中，且该标签唯一。然后再按照 el 的类提取出一个列表，每个列表中 ti（i = 1,2,3,4,5）都为目标数据。其中要注意的是。第一个 el 的‘t1’较为特殊，是‘t1 tg1’，因此使用 scrapy 的 css 选择器解析时并不方便，会多好几行代码处理这个特殊情况，故选择了可以模糊检索的 BeautifulSoup。（事实上，在项目开始的两天之后服务器已经把这个奇怪的类修改回‘t1’）

```
▼<div class="dw_table" id="resultList">
  <!-- 关键字广告 start -->
  <!-- 关键字广告 end -->
  <div id="dw_tlc_mk"></div>
 ▶<div class="dw_tlc">…</div>
  <!--列表表格 start-->
 ▶<div class="el title">…</div>
 ▼<div class="el">
  ▼<p class="t1 ">
     <em class="check" name="delivery_em" onclick="checkboxClick(this)"></em>
     <input class="checkbox" type="checkbox" name="delivery_jobid" value="123172979" jt="0" style="display:none">
   ▼<span>
      <a target="_blank" title="Python开发工程师" href="https://jobs.51job.com/shanghai-xhq/123172979.html?s=01&t=0" onmousedown>
                        Python开发工程师                </a>
     </span>
   </p>
  ▼<span class="t2">
     <a target="_blank" title="杉德巍康企业服务有限公司" href="https://jobs.51job.com/all/co2188232.html">杉德巍康企业服务有限公司</a>
   </span>
   <span class="t3">上海-徐汇区</span>
   <span class="t4">1-1.5万/月</span>
   <span class="t5">06-25</span>
 </div>
```

（3）薪资文本处理：

在分析数据的时候发现。有很多工作招聘信息并没有直接给出工资范围。而是选择面议。因此在网页中会出现薪资一栏是空白的情况，于是在文本处理时，这一类只好做 0 处理。

Python开发工程师	北京金信润天信息技术股份有限公司...	武汉-洪山区	0.7-1.3万/月	06-25
Python开发工程师	上海期货信息技术有限公司	上海-浦东新区		06-25

除去特殊的面议工资问题外。51job 给出的工资形式分为两种：

a. X 元/天

b. X-Y[元千万]/[年月]

因此在文本处理时，先要对 ab 两种情形进行判定。然后再分拆计算。

（4）scrapy 配置文件设置：

由于 scrapy 的框架具有一定的固定性。因此不如 requests 灵活。不同的目标只有通过 settings.py 配置文件来实现。在本项目中，额外增添了 MAX_PAGECRAWL_NUMBER 用于控制爬取的页面数量，修改了 USER_AGENT 伪装 header。设置 ROBOTSTXT_OBEY 为 False 确保绕过 51job 的 ROBOT 协议完成数据爬取工作以及控制台信息输出。还新写了 JobInfoPipeline 用于处理数据，修改了默认管道配置。

（5）数据库编写：

数据库编写秉持一切从简的目的。在之前作业中编写的 SQL 类的基础做了大量的删改：原先的 SQL 的增删改查遍历等功能尽数摈弃参数输入，只保留了最基本的功能供 Pipeline 调用输入数据。

3.4 鲁棒性分析

事实上，利用 scrapy 框架编写的爬虫程序比 requests 程序速度提高了一倍以上。但是带来了一定的不稳定因素。由于 scrapy 爬取速度过快，即便在配置文件里设置 downloader 延迟为 3 秒也依然有概率被服务器判定为频繁访问的恶意攻击。根据实验发现，出现被服务器强制中断链接问题的原因主要是短时间多次启动爬虫。只要注意是用频率就不会出现该问题。

事实上，原本打算通过代理 IP 和 IP 池的方式来解决 IP 封禁问题。实验发现，51job 的技术手段仅仅是在一段时间之内不允许恶意 IP 的连接。因此只要过一段时间再次启动程序就可以正常运行了。

四、课程设计总结

4.1 项目检查方法

本项目保存在 https://github.com/WOWspring/pythonhomework.git 中的“exe11\Problem7”文件夹下。所需要的配置环境可用 requirements.txt 文件直接 pip 安装。所需要的数据库可以使用 job_search.sql 文件自动生成（该文件中还保存了一次爬取记录）。

使用该爬虫时，进入到“exe11\Problem7\findjob_python\findjob_python”文件夹下，运行 start.py 后，首先输入想要查询的工作，之后输入数据库密码后即可爬取网页数据并同步保存到数据库和文本文档。期间因为 scrapy 爬虫速度过快的原因可能会被当成恶意攻击导致爬取暂时中断。耐心等候服务器会自动放开该 IP 从而继续爬取。

爬取完成之后进入“exe11\Problem7\findjob_python\findjob_python\parse”执行 count_average_salary.py 文件。该程序会通过 data 文件夹下的 JobInfo.txt 计算北京地区的平均薪资。之后可以按照控制台文本提示继续查找计算其他地区的平均薪资或者退出程序。

4.2 总结及体会

本次项目是完全在 Pycharm 平台，使用 Python 语言开发的。使用了市面上较为成熟的 scrapy 框架，并且通过查阅 scrapy 的 api 文档完成了项目。固然在项目开发中遇到了框架不熟悉，api 阅读困难，git 提交错误等问题。但都通过查阅资料等方法解决了。通过这次课设，不仅提高了查阅资料的水平，更熟悉了解了生成器，迭代器等 python 难点。虽然有多篇论文和课设同步进行导致时间有所欠缺，但还是按时完成了既定目标。

参考文献

[1]Scrapy.Scrapy 2.1 documentation — Scrapy 2.1.0 documentation[CP/DK].https://docs.scrapy.org/en/latest/,2020-04-24.