

链家全国房源信息千万级全站爬虫（可断点续传，邮件通知）

想做一些房产类数据分析的工作，想找一下比较大体量的数据爬虫，csdn找了一圈不是收费就是培训机构打广告的，有些付费的还是一些单页爬虫只能拿到几十条也好意思收费，github上维护的比较少，都是前几年的代码了，网页结构发生了变化，程序自然也就跑不了，既然找不到那就干脆自己动手吧！

思路分析：

想淘宝京东一样，定位到某一个城市链家默认展示出来的页码最多只有100页，也就是3000套，对于一些大城市的房源数量是可以达到十万级的，如果只是简单的做一个翻页爬取，那么获得的数据量相比实际值差别太大，也会影响到数据分析的准确性。我们不妨，在选择到一个城市之后我们可以细分到其下属的行政区和街道。如果我们能够顺利地获取到城市的每个街道的房源数（前面说到最大的展示页码是100）那么我们所得到的数据量会多得多。既然如此，干脆从把全国的房源信息都爬下来吧。首先我们需要拿到城市--区域--街道--房源数量的关系。然后通过遍历这些关系实现对链家的全站爬取。我这里采用MySQL作为一个中间表，先获取全国城市链接--再获取每个城市下的所有行政区链接--然后获取每个行政区下的所有街道链接--最后获取每个街道链接下的房源数量来决定每个街道应该翻多少页。通过这几个步骤我们就能获取到最后的数据集。

准备工作：

- 1.这里用到了MySQL数据库，版本是8的，这个请自行安装
- 2.中间表的爬取由于涉及到多页面跳转这里就使用selenium模块配合谷歌浏览器
- 3.语言用python，编译器是pycharm
- 4.用到的一些库，没有的可以pip一下

```
import datetime
import sys
from time import sleep
import pymysql
# 导入pyquery解析页面
from selenium import webdriver
# 让selenium规避被检测风险
from selenium.webdriver import ChromeOptions
from selenium.webdriver.common.by import By
# 导入显示等待参数
from selenium.webdriver.support.ui import WebDriverWait
# 捕获抓不到元素和请求超时的异常
from selenium.common.exceptions import NoSuchElementException, TimeoutException
```

一.获取全国所有城市信息名称和URL

链家全国城市页面：<https://www.lianjia.com/city/>

安徽	安庆	滁州	阜阳	合肥	马鞍山	芜湖
北京	北京					
重庆	重庆					
福建	福州	泉州	厦门	漳州		
广东	东莞	佛山	广州	惠州	江门	清远
	深圳	珠海	湛江			
广西	北海	防城港	桂林	柳州	南宁	
贵州	贵阳	黔东南				

分析上面的页面我们需要获取的有省份，城市名，以及城市的url

```

▼ <ul class="city_list_ul"> == $0
  ▶ <li class="city_list_li city_list_li_selected">...</li>
  ▶ <li class="city_list_li city_list_li_selected">...</li>
  ▶ <li class="city_list_li city_list_li_selected">...</li>
  ▶ <li class="city_list_li city_list_li_selected">...</li>
  ▶ <li class="city_list_li city_list_li_selected">...</li>
  ▶ <li class="city_list_li city_list_li_selected">...</li>
  ▶ <li class="city_list_li city_list_li_selected">...</li>
  ▶ <li class="city_list_li city_list_li_selected">...</li>

```

首先抓到这些元素所在的标签：items = bro.find_elements(By.XPATH, 'html/body/div[2]/div[2]/div/div/ul/li')

那么第一个我们找省份：但是省份是在不同的首字母板块之下的，我们需要对所有的字母板块遍历才能拿到所有的省份。

```

for item in items:
    # 首字母板块基础取省份
    for i in item.find_elements(By.XPATH, './div[2]/div'):
        '省份': i.find_element(By.XPATH, './div').text

```

同理城市又是在省份之下

```

# 首字母板块基础取省份
for i in item.find_elements(By.XPATH, './div[2]/div'):
    # 省份基础取城市
    for u in i.find_elements(By.XPATH, './ul/li'):
        '城市': u.text

```

最后是城市的链接（这里直接拼上一个ershoulfang就可以直达每个城市的二手房列表）


```
}
print(city_info)
```

效果图：

省份	城市	城市主键	城市链接	当前信息获取时间	当前城市下属区域是否已获取
安徽	安庆	aq	https://aq.lianjia.com/ershoufang	2022-02-23 17:33:01	0
陕西	宝鸡	baoji	https://baoji.lianjia.com/ershoufang	2022-02-23 17:33:10	0
内蒙古	包头	baotou	https://baotou.lianjia.com/ershoufang	2022-02-23 17:33:08	0
河北	保定	bd	https://bd.lianjia.com/ershoufang	2022-02-23 17:33:03	0
广西	北海	bh	https://bh.lianjia.com/ershoufang	2022-02-23 17:33:02	0
北京	北京	bj	https://bj.lianjia.com/ershoufang	2022-02-23 17:33:01	0
海南	保亭	bt.fang	https://bt.fang.lianjia.com/ershoufang	2022-02-23 17:33:04	0
内蒙古	巴彦淖尔	by.fang	https://by.fang.lianjia.com/ershoufang	2022-02-23 17:33:08	0

最后在完整代码中再放出PySQL的入库方法。

二.获取每个城市下所有行政区名和所对应的URL

这里就需要遍历上面获得的全国城市表来获取行政区

区域表设计：

字段	索引	外键	触发器	选项	注释	SQL 预览
名					类型	长度 小数点 不是 null 虚拟 键 注释
省份					varchar	255 <input type="checkbox"/> <input type="checkbox"/>
城市					varchar	255 <input type="checkbox"/> <input type="checkbox"/>
区域					varchar	255 <input type="checkbox"/> <input type="checkbox"/>
区域主键					varchar	255 <input checked="" type="checkbox"/> <input type="checkbox"/> 1
区域链接					varchar	255 <input checked="" type="checkbox"/> <input type="checkbox"/>
当前信息获取时间					varchar	255 <input type="checkbox"/> <input type="checkbox"/>
当前区域下属街道是否已获取					int	<input checked="" type="checkbox"/> <input type="checkbox"/>

方法代码：

```
def get_area():
    # 使用cursor()方法获取操作游标
    cursor = DB.cursor()
    sql = "select 省份,城市,城市链接 from " + '城市中间表'
    cursor.execute(sql)
    info = cursor.fetchall()
    DB.commit()
    for i in info:
        # 这里取出的每个i是元组类型,如果需要获取城市链接应当用i[2]来表示
        try:
            bro.get(i[2])
        except TimeoutException:
            print('=====连接超时, 10秒后重试=====')
            try:
                sleep(10)
                bro.get(i[2])
            except TimeoutException:
                print('=====连接超时, 请重新运行=====')
                bro.quit()
                sys.exit()
        try:
            items = bro.find_elements(By.XPATH,
                '/html/body/div[3]/div/div[1]/dl[2]/dd/div[1]/div[1]/a')
        except NoSuchElementException:
```

```

print('=====未能找到指定元素，可能遇到爬虫
=====')
bro.quit()
sys.exit()
for item in items:
    area_info = {
        '省份': i[0],
        '城市': i[1],
        '区域': item.text,
        '区域主键': item.get_attribute('href').replace('https://',
        '').replace('.lianjia.com/ershoufang/',
        '').replace('/', ''),
        '区域链接': item.get_attribute('href'),
        '当前信息获取时间': datetime.datetime.now().strftime('%Y-%m-%d
%H:%M:%S'),
        '当前区域下属街道是否已获取': '0'
    }
    print(area_info)
cursor.close()

```

效果图：

省份	城市	区域	区域主键	区域链接	当前区域下属街道是否已获取
安徽	安庆	大观区	aqdaguanqu	https://aq.lianjia.com/ershoufan	1
安徽	安庆	怀宁县	aqhuainingxian	https://aq.lianjia.com/ershoufan	1
安徽	安庆	潜山县	aqqianshanxian	https://aq.lianjia.com/ershoufan	1
安徽	安庆	宿松县	aqsusongxian	https://aq.lianjia.com/ershoufan	1
安徽	安庆	太湖县	aqtaihuxian	https://aq.lianjia.com/ershoufan	1
安徽	安庆	桐城市	aqtongchengshi	https://aq.lianjia.com/ershoufan	1

三.获取每个行政区下所有街道名和所对应的URL

全国所有街道信息已经达到7300+了，那么我们就需要对已经获取玩所有街道的行政区表做一个标志，把原来的标识符0置为1，那么哪怕程序中断或者网络不好再运行的情况下也只会去获取未获取所有街道的行政区，同理因为设计了主键，是会出现重复数据的。

```

# # 当前信息已经解析出更细化的信息时，把标志字段重置为1
sql_update = 'UPDATE 区域中间表 SET 当前区域下属街道是否已获取 = %s WHERE 区域主
键 = %s'
try:
    cursor.execute(sql_update, (1, i[3]))
    DB.commit()
except pymysql.Error as e:
    DB.rollback()
    print("Mysql Error %d: %s" % (e.args[0], e.args[1]))

```

表设计：

保存

添加字段


插入字段

删除字段

主键

上移

下移

字段	索引	外键	触发器	选项	注释	SQL 预览				
名				类型	长度	小数点	不是 null	虚拟	键	注释
省份				varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
城市				varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
区域				varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
街道				varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
街道主键				varchar	255		<input checked="" type="checkbox"/>	<input type="checkbox"/>	 1	
街道链接				varchar	255		<input checked="" type="checkbox"/>	<input type="checkbox"/>		
当前信息获取时间				varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
当前街道下房源数量是否已获取				int			<input checked="" type="checkbox"/>	<input type="checkbox"/>		

方法代码：

```

# 获取每个区域下的街道信息
def get_street():
    # 使用cursor()方法获取操作游标
    cursor = DB.cursor()
    # 只取未获取的区域来执行
    sql = "select 省份,城市,区域,区域主键,区域链接 from 区域中间表 WHERE 当前区域下属街道是否已获取 = 0"
    cursor.execute(sql)
    info = cursor.fetchall()
    DB.commit()
    for i in info:
        # 这里取出的每个i是元组类型,如果需要获取城市链接应当用i[2]来表示
        try:
            bro.get(i[4])
        except TimeoutException:
            print('=====连接超时, 10秒后重试=====')
            try:
                sleep(10)
                bro.get(i[4])
            except TimeoutException:
                print('=====连接超时, 请检查网络=====')
            bro.quit()
            sys.exit()

        # 定位到街道url TODO @href不能解析多个属性, 需要用到get_attribute方法
        try:
            items = bro.find_elements(By.XPATH,
                '/html/body/div[3]/div/div[1]/dl[2]/dd/div[1]/div[2]/a')
        except NoSuchElementException:
            print('=====未能找到指定元素, 可能遇到爬虫=====')
            bro.quit()
            # 未能解析到信息就把进程中止, 防止出现把没有获取到的房源做了错误的标记
            sys.exit()

        for item in items:
            street_info = {
                '省份': i[0],
                '城市': i[1],
                '区域': i[2],
                '街道': item.text,
                '街道主键': item.get_attribute('href').replace('https://',
                    '').replace('.lianjia.com/ershoufang/'),
                '': item.replace('/', ''),
            }

```

```

        '街道链接': item.get_attribute('href'),
        '当前信息获取时间': datetime.datetime.now().strftime('%Y-%m-%d
%H:%M:%S'),
        '当前街道下房源数量是否已获取': '0'
    }
    print(street_info)
    save_to_mysql('街道中间表', street_info)
    ## 当前信息已经解析出更细化的信息时,把标志字段重置为1
    sql_update = 'UPDATE 区域中间表 SET 当前区域下所属街道是否已获取 = %s WHERE 区域主
键 = %s'
    try:
        cursor.execute(sql_update, (1, i[3]))
        DB.commit()
    except pymysql.Error as e:
        DB.rollback()
        print("Mysql Error %d: %s" % (e.args[0], e.args[1]))
    cursor.close()

```

效果图:

省份	城市	区域	街道	街道主键	街道链接	当前当前街道下房源数量是否已获取
安徽	安庆	宜秀区	北部新城	aqbeibuxincheng	https://aq.lianjia.com/ershoufang/beibuxincheng/	202. 1
安徽	安庆	宜秀区	大桥	aqdaqiao	https://aq.lianjia.com/ershoufang/daqiao/	202. 1
安徽	安庆	迎江区	德宽路	aqdekuanlu	https://aq.lianjia.com/ershoufang/dekuanlu/	202. 1
安徽	安庆	宜秀区	怀宁县	aqhuainingxian1	https://aq.lianjia.com/ershoufang/huainingxian1/	202. 1
安徽	安庆	宜秀区	花亭	aqhuating1	https://aq.lianjia.com/ershoufang/huating1/	202. 1
安徽	安庆	迎江区	华中路	aqhuazhonglu	https://aq.lianjia.com/ershoufang/huazhonglu/	202. 1
安徽	安庆	宜秀区	老峰镇	aqlaofengzhen	https://aq.lianjia.com/ershoufang/laofengzhen/	202. 1

四.获取每个街道下所有房源数

至此已经获取到全国的街道信息和对应url,不妨我们直接再去遍历街道信息表得到每个街区对应的房源数量以及对应需要的翻页数那么在后面就能更快速的爬取到我们的信息。

表设计:

名	类型	长度	小数点	不是 null	虚拟	键	注释
省份	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
城市	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
区域	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
街道	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
街道主键	varchar	255		<input checked="" type="checkbox"/>	<input type="checkbox"/>	1	
街道链接	varchar	255		<input checked="" type="checkbox"/>	<input type="checkbox"/>		
当前街道房源数量	varchar	255		<input checked="" type="checkbox"/>	<input type="checkbox"/>		
当前街道房源页数	varchar	255		<input checked="" type="checkbox"/>	<input type="checkbox"/>		
当前信息获取时间	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
当前街道房源是否全部已获取	int			<input type="checkbox"/>	<input type="checkbox"/>		

我们只能获取到房源数,需要自己写逻辑判断出页数,每页30条,自己分析一下就可以了,下面给代码

```

if int(num) == 0:
    page = 0
elif 0 < int(num) < 3000:
    if int(num) % 30 == 0:
        page = (int(num) // 30)
    elif int(num) % 30 != 0:
        page = (int(num) // 30 + 1)
elif int(num) >= 3000:
    page = int(100)

```

方法代码:

```
# 获取最小单位街道下的所以房源数量
def get_number():
    num = ''
    page = ''
    cursor = DB.cursor()
    # 只取未获取的街道来执行
    sql = "select 省份,城市,区域,街道,街道主键,街道链接 from 街道中间表 WHERE 当前街道下属
    房源数量是否已获取 = 0"
    cursor.execute(sql)
    info = cursor.fetchall()
    DB.commit()
    for i in info:
        # 这里取出的每个i是元组类型,如果需要获取城市链接应当用i[2]来表示
        try:
            bro.get(i[5])
        except TimeoutException:
            print('=====连接超时, 10秒后重试=====')
            try:
                sleep(10)
                bro.get(i[5])
            except TimeoutException:
                print('=====连接超时, 请重新运行
                =====')
                bro.quit()
                sys.exit()

        try:
            num = bro.find_element(By.XPATH, '//*[@id="content"]/div[1]/div[2]/h2/span').text
            if int(num) == 0:
                page = 0
            elif 0 < int(num) < 3000:
                if int(num) % 30 == 0:
                    page = (int(num) // 30)
                elif int(num) % 30 != 0:
                    page = (int(num) // 30 + 1)
            elif int(num) >= 3000:
                page = int(100)
        except NoSuchElementException:
            print('=====未能找到指定元素, 可能遇到爬虫
            =====')
            # 未能解析到信息就把进程中止, 防止出现把没有获取到的房源做了错误的标记
            bro.quit()
            sys.exit()

    number_info = {
        '省份': i[0],
        '城市': i[1],
        '区域': i[2],
        '街道': i[3],
        '街道主键': i[4],
        '街道链接': i[5],
        '当前街道房源数量': num,
        '当前街道房源页数': page,
        '当前信息获取时间': datetime.datetime.now().strftime('%Y-%m-%d
        %H:%M:%S'),
        '当前街道房源是否全部已获取': '0'
```



```

    }
    print(number_info)
    save_to_mysql('房源数量中间表', number_info)
    # # 当前信息已经解析出更细化的信息时, 把标志字段重置为1
    sql_update = 'UPDATE 街道中间表 SET 当前街道下房源数量是否已获取 = %s WHERE 街道主键 = %s'
    try:
        cursor.execute(sql_update, (1, i[4]))
        DB.commit()
    except pymysql.Error as e:
        DB.rollback()
        print("Mysql Error %d: %s" % (e.args[0], e.args[1]))

cursor.close()

```

效果图:

省份	城市	区域	街道	街道主键	街道链接	当前街道房源数量	当前街道房源页数	当前信息获取时间	当前街道房源是否全部已获取
安徽	安庆	宜秀区	北部新城	aqbeibuxincheng	https://aq.li470	16	2022-02-25 19:39:57		1
安徽	安庆	宜秀区	大桥	aqdaqiao	https://aq.li2992	100	2022-02-25 19:40:00		1
安徽	安庆	迎江区	德宽路	aqdekuanlu	https://aq.li843	29	2022-02-25 19:40:01		1
安徽	安庆	宜秀区	怀宁县	aqhuainingxian1	https://aq.li1322	45	2022-02-25 19:40:03		1
安徽	安庆	宜秀区	花亭	aqhuating1	https://aq.li428	15	2022-02-25 19:40:05		1
安徽	安庆	迎江区	华中路	aqhuazhonglu	https://aq.li1233	42	2022-02-25 19:40:07		1
安徽	安庆	宜秀区	老峰镇	aqiaofengzhen	https://aq.li500	17	2022-02-25 19:40:09		1
安徽	安庆	宜秀区	菱北	aqqingbei	https://aq.li4166	100	2022-02-25 19:40:11		1

所有中间表获取的整体代码:

可以参考一下入库和取数方法以及页面解析方法, 都有详细注释, 还可以参考一下异常捕获, 欢迎大家提出宝贵意见。

```

import datetime
import sys
from time import sleep
import pymysql
# 导入pyquery解析页面
from selenium import webdriver
# 让selenium规避被检测风险
from selenium.webdriver import ChromeOptions
from selenium.webdriver.common.by import By
# 导入显示等待参数
from selenium.webdriver.support.ui import WebDriverWait
# 捕获抓不到元素和请求超时的异常
from selenium.common.exceptions import NoSuchElementException, TimeoutException

# 获取链家全国城市链接方法
def get_city():
    try:
        bro.get('https://www.lianjia.com/city/')
    except TimeoutException:
        print('=====连接超时, 10秒后重试=====')
        try:
            sleep(10)
            bro.get('https://www.lianjia.com/city/')
        except TimeoutException:
            print('=====连接超时, 请检查网络=====')
            bro.quit()

```

```

        sys.exit()

    try:
        items = bro.find_elements(By.XPATH,
'/html/body/div[2]/div[2]/div/div/ul/li')
    except NoSuchElementException:
        print('=====未能找到指定元素，可能遇到爬虫
=====')
        bro.quit()
        sys.exit()

# 网页基础取首字母板块
for item in items:
    # 首字母板块基础取省份
    for i in item.find_elements(By.XPATH, './div[2]/div'):
        # 省份基础取城市
        for u in i.find_elements(By.XPATH, './ul/li'):
            city_info = {
                # 这里一定是用find_element方法而不是find_elements
                '省份': i.find_element(By.XPATH, './div').text,
                # 这里一定是用find_element方法而不是find_elements
                '城市': u.text,
                '城市主键': u.find_element(By.XPATH,
'/a').get_attribute('href').replace('https://', '').replace(
                '.lianjia.com/', ''),
                '城市链接': u.find_element(By.XPATH,
'/a').get_attribute('href') + 'ershoufang/',
                '当前信息获取时间': datetime.datetime.now().strftime('%Y-%m-%d
%H:%M:%S'),

                '当前城市下属区域是否已获取': '0'
            }
            print(city_info)
            save_to_mysql('城市中间表', city_info)

# 获取每个城市下的全部区域信息
def get_area():
    # 使用cursor()方法获取操作游标
    cursor = DB.cursor()
    sql = "select 省份,城市,城市链接 from " + '城市中间表'
    cursor.execute(sql)
    info = cursor.fetchall()
    DB.commit()
    for i in info:
        # 这里取出的每个i是元组类型,如果需要获取城市链接应当用i[2]来表示
        try:
            bro.get(i[2])
        except TimeoutException:
            print('=====连接超时，10秒后重试=====')
            try:
                sleep(10)
                bro.get(i[2])
            except TimeoutException:
                print('=====连接超时，请重新运行
=====')
                bro.quit()
                sys.exit()

    try:
        items = bro.find_elements(By.XPATH,
'/html/body/div[3]/div/div[1]/dl[2]/dd/div[1]/div[1]/a')

```

```

        except NoSuchElementException:
            print('=====未能找到指定元素，可能遇到爬虫
=====')
            bro.quit()
            sys.exit()
        for item in items:
            area_info = {
                '省份': i[0],
                '城市': i[1],
                '区域': item.text,
                '区域主键': item.get_attribute('href').replace('https://',
'').replace('.lianjia.com/ershoufang/',
'').replace('/', ''),
                '区域链接': item.get_attribute('href'),
                '当前信息获取时间': datetime.datetime.now().strftime('%Y-%m-%d
%H:%M:%S'),
                '当前区域下属街道是否已获取': '0'
            }
            print(area_info)
            save_to_mysql('区域中间表', area_info)
        cursor.close()

# 获取每个区域下的街道信息
def get_street():
    # 使用cursor()方法获取操作游标
    cursor = DB.cursor()
    # 只取未获取的区域来执行
    sql = "select 省份,城市,区域,区域主键,区域链接 from 区域中间表 WHERE 当前区域下属街道
是否已获取 = 0"
    cursor.execute(sql)
    info = cursor.fetchall()
    DB.commit()
    for i in info:
        # 这里取出的每个i是元组类型,如果需要获取城市链接应当用i[2]来表示
        try:
            bro.get(i[4])
        except TimeoutException:
            print('=====连接超时，10秒后重试=====')
            try:
                sleep(10)
                bro.get(i[4])
            except TimeoutException:
                print('=====连接超时，请检查网络
=====')
            bro.quit()
            sys.exit()
        # 定位到街道url TODO @href不能解析多个属性，需要用到get_attribute方法
        try:
            items = bro.find_elements(By.XPATH,
'/html/body/div[3]/div/div[1]/dl[2]/dd/div[1]/div[2]/a')
        except NoSuchElementException:
            print('=====未能找到指定元素，可能遇到爬虫
=====')
            bro.quit()
            # 未能解析到信息就把进程中止，防止出现把没有获取到的房源做了错误的标记
            sys.exit()

```

```

        for item in items:
            street_info = {
                '省份': i[0],
                '城市': i[1],
                '区域': i[2],
                '街道': item.text,
                '街道主键': item.get_attribute('href').replace('https://',
'').replace('.lianjia.com/ershoufang/',
'').replace('/', ''),
                '街道链接': item.get_attribute('href'),
                '当前信息获取时间': datetime.datetime.now().strftime('%Y-%m-%d
%H:%M:%S'),
                '当前街道下属房源数量是否已获取': '0'
            }
            print(street_info)
            save_to_mysql('街道中间表', street_info)
        # # 当前信息已经解析出更细化的信息时, 把标志字段重置为1
        sql_update = 'UPDATE 区域中间表 SET 当前区域下属街道是否已获取 = %s WHERE 区域主
键 = %s'
        try:
            cursor.execute(sql_update, (1, i[3]))
            DB.commit()
        except pymysql.Error as e:
            DB.rollback()
            print("Mysql Error %d: %s" % (e.args[0], e.args[1]))
        cursor.close()

# 获取最小单位街道下的所以房源数量
def get_number():
    num = ''
    page = ''
    cursor = DB.cursor()
    # 只取未获取的街道来执行
    sql = "select 省份,城市,区域,街道,街道主键,街道链接 from 街道中间表 WHERE 当前街道下属
房源数量是否已获取 = 0"
    cursor.execute(sql)
    info = cursor.fetchall()
    DB.commit()
    for i in info:
        # 这里取出的每个i是元组类型, 如果需要获取城市链接应当用i[2]来表示
        try:
            bro.get(i[5])
        except TimeoutException:
            print('=====连接超时, 10秒后重试=====')
            try:
                sleep(10)
                bro.get(i[5])
            except TimeoutException:
                print('=====连接超时, 请重新运行
=====')
                bro.quit()
                sys.exit()

        try:
            num = bro.find_element(By.XPATH, '//*[@id="content"]/div[1]/div[2]/h2/span').text
            if int(num) == 0:

```

```

        page = 0
    elif 0 < int(num) < 3000:
        if int(num) % 30 == 0:
            page = (int(num) // 30)
        elif int(num) % 30 != 0:
            page = (int(num) // 30 + 1)
    elif int(num) >= 3000:
        page = int(100)
except NoSuchElementException:
    print('=====未能找到指定元素，可能遇到爬虫
=====')
    # 未能解析到信息就把进程中止，防止出现把没有获取到的房源做了错误的标记
    bro.quit()
    sys.exit()
number_info = {
    '省份': i[0],
    '城市': i[1],
    '区域': i[2],
    '街道': i[3],
    '街道主键': i[4],
    '街道链接': i[5],
    '当前街道房源数量': num,
    '当前街道房源页数': page,
    '当前信息获取时间': datetime.datetime.now().strftime('%Y-%m-%d
%H:%M:%S'),
    '当前街道房源是否全部已获取': '0'
}
print(number_info)
save_to_mysql('房源数量中间表', number_info)
# # 当前信息已经解析出更细化的信息时，把标志字段重置为1
sql_update = 'UPDATE 街道中间表 SET 当前街道下房源数量是否已获取 = %s WHERE 街
道主键 = %s'
try:
    cursor.execute(sql_update, (1, i[4]))
    DB.commit()
except pymysql.Error as e:
    DB.rollback()
    print("Mysql Error %d: %s" % (e.args[0], e.args[1]))

cursor.close()

# key是各个中间表名,result是各个中间表结果
def save_to_mysql(key, result):
    cursor = DB.cursor()
    table = str(key)
    keys = ', '.join(result.keys())
    values = ', '.join(['%s' * len(result)])

    sql = 'INSERT INTO {table}({keys}) VALUES ({values}) ON DUPLICATE KEY
UPDATE'.format(table=table, keys=keys,

    values=values)
    update = ', '.join([" {key} = %s".format(key=key) for key in result])
    sql += update
    try:
        if cursor.execute(sql, tuple(result.values()) * 2):
            print('存储到MySQL数据库成功')

```

```

        DB.commit()
    except pymysql.Error as e:
        DB.rollback()
        print("Mysql Error %d: %s" % (e.args[0], e.args[1]))
    finally:
        cursor.close()

def main():
    # 1.先获取全国所有城市信息
    print('=====开始获取链家全国所有城市信息
=====')
    # get_city()
    print('=====链家全国所有城市信息获取完成
=====\\n')
    # 2.获取每个城市全部区域信息
    print('=====开始获取链家全国所有区域信息
=====')
    # get_area()
    print('=====链家全国所有区域信息获取完成
=====\\n')
    # 3.获取每个区域全部街道信息
    print('=====开始获取链家全国所有街道信息
=====')
    # get_street()
    print('=====链家全国所有街道信息获取完成
=====\\n')
    # 4.获取全部街道下所有房源数
    print('=====开始获取链家全国所有街道房源数量
=====')
    # get_number()
    print('=====链家全国所有街道房源数量获取完成
=====')

if __name__ == '__main__':
    # 初始化MySQL
    try:
        DB = pymysql.connect(host='localhost', port=3307, user='root',
password='000000', database='house')
    except pymysql.Error as e:
        print("Mysql Error %d: %s" % (e.args[0], e.args[1]))
        print('=====请启动Mysql8=====')
    # 设置浏览器参数对象
    option = ChromeOptions()
    # 关闭chrome弹出得密码保存框和网页通知询问
    prefs = {"": "", "credentials_enable_service": False,
"profile.password_manager_enabled": False,
"profile.default_content_setting_values": {'notifications': 2}}
    option.add_experimental_option("prefs", prefs)
    # 关闭自动化测试提醒
    option.add_experimental_option("excludeSwitches", ['enable-automation'])
    # 屏蔽webdriver特征
    option.add_argument("--disable-blink-features")
    option.add_argument("--disable-blink-features-AutomationControlled")
    # TODO 使用无头
    option.add_argument('--headless')
    # 初始化引擎参数

```

```

bro = webdriver.Chrome(options=option)
# 反淘宝js检测手段 解决滑块验证码
bro.execute_cdp_cmd("Page.addScriptToEvaluateOnNewDocument", {
    "source": "Object.defineProperty(navigator, 'webdriver', {get: () =>
undefined})",
})
# 配置显示等待参数传入引擎对象和最大等待时间
wait = WebDriverWait(bro, 60)
print('=====开始获取全国所有房源信息
=====')
main()
sleep(3)
bro.quit()
DB.close()

```

到此为止，所有中间表都已经完成，其实整体实现就是按照一开始的思路设计，遍历上一个表写入下一个表，一层一层地从城市到区域到街道到所有的房源数量以及url，那么我们只需要简单的去遍历最后一张房源数量中间表，就可以解析出我们所需要的信息。

下面就可以直接获取房源基本信息了。

表设计：

名	类型	长度	小数点	不是 null	虚拟	键	注释
房源主键	varchar	255		<input checked="" type="checkbox"/>	<input type="checkbox"/>	1	
省份	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
城市	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
所属行政区	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
所在街道	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
小区名称	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
价格	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
建筑面积	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
每平方单价	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
户型	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
装修情况	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
房屋朝向	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
楼层	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
楼龄	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
建筑类型	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
发布时间	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
关注人数	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
房源标签	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
房源链接	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
房源图片链接	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
当前信息时间	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
是否已获取该房源明细信息	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		

整体房源基本信息代码：

```

import datetime
import sys
from time import sleep

import pymysql
import requests
# 导入pyquery解析页面
from pyquery import PyQuery as pq
from selenium import webdriver

```

```

# 让selenium规避被检测风险
from selenium.common.exceptions import NoSuchElementException, TimeoutException
from selenium.webdriver import ChromeOptions
# 导入显示等待参数
from selenium.webdriver.common.by import By

def request(province_1, city_1, area_1, link_1):
    # 设置请求头
    headers = {
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) '
        'Chrome/98.0.4758.102 Safari/537.36 Edg/98.0.1108.55',
        'Referer': 'https://lianjia.com/',
    }
    # 请求经过判断处理后的url
    try:
        response = requests.get(url=link_1, headers=headers, timeout=6)
    except requests.exceptions.RequestException as e:
        print('=====网络不通，重试一次')
        print('=====')
        try:
            sleep(10)
            response = requests.get(url=link_1, headers=headers, timeout=60)
        except requests.exceptions.RequestException as e:
            print('=====网络不通，请重新运行')
            print('=====')
            sys.exit()
    # TODO 再做一个监测反爬虫的
    # 响应码是200而不是'200'
    if '封禁原因' not in response.text:
        analysis(province_1, city_1, area_1, response)
    else:
        print('=====存在反爬虫，请重新运行')
        print('=====')
        bro.quit()
        sys.exit()

def analysis(province_2, city_2, area_2, response):
    # TODO 选择request或者browser方法还要对应选择这里

    # request
    doc = pq(response.text)

    # browser
    # doc = pq(response)

    # items = doc('.sellListContent li').items
    # TODO 改用下面写法指定第一个元素可以避免找到小于30套房源的广告信息 find是找子孙
    # children是找子节点
    items = doc.find('.sellListContent').eq(0).children('li').items()
    for item in items:
        # 因为这个六个属性都在一个元素标签下，需要经过切分才能细分属性，但是房源元素不统一需要
        # 做一个判断
        # huxing = ''
        # mianzhi = ''
        # chaoxiang = ''

```



```

# zhuangxiu = ''
# loucheng = ''
# louling = ''
# jianzhuleixing = ''
# TODO 去掉车位信息
chewei = item.find('.houseInfo').text().replace(' ', '').split('|')[0]
# 判断class=houseInfo下面的标签个数是否等于7，如果不等于7直接取值到字典会造成数据混
乱，所以当前
if len(item.find('.houseInfo').text().split('|')) == 7:
    huxing = item.find('.houseInfo').text().replace(' ', '').split('|')
[0]
    mianzhi = item.find('.houseInfo').text().replace(' ', '').split('|')
[1]
    chaoxiang = item.find('.houseInfo').text().replace(' ',
''.split('|')[2]
    zhuangxiu = item.find('.houseInfo').text().replace(' ',
''.split('|')[3]
    loucheng = item.find('.houseInfo').text().replace(' ',
''.split('|')[4]
    louling = item.find('.houseInfo').text().replace(' ', '').split('|')
[5]
    jianzhuleixing = item.find('.houseInfo').text().replace(' ',
''.split('|')[6]
    # TODO 目前分出6个标签的房源和7个标签的房源 后续根据数据样本进一步细化，提高数据
质量
elif len(item.find('.houseInfo').text().split('|')) == 6:
    huxing = item.find('.houseInfo').text().replace(' ', '').split('|')
[0]
    mianzhi = item.find('.houseInfo').text().replace(' ', '').split('|')
[1]
    chaoxiang = item.find('.houseInfo').text().replace(' ',
''.split('|')[2]
    zhuangxiu = item.find('.houseInfo').text().replace(' ',
''.split('|')[3]
    loucheng = item.find('.houseInfo').text().replace(' ',
''.split('|')[4]
    louling = '暂无数据'
    jianzhuleixing = jianzhuleixing =
item.find('.houseInfo').text().replace(' ', '').split('|')[-1]
else:
    # 房源信息的7个属性均设为暂无数据，虽然缺失了一部分数据，但提高了整体数据完整度
    huxing = mianzhi = chaoxiang = zhuangxiu = loucheng = louling =
jianzhuleixing = '暂无数据'

house_info = {
    '房源主键': item.find('.title a').attr('href').replace('https://',
'').replace('.lianjia.com/ershoufang/',
'').replace('.html', ''),
    '省份': province_2,
    '城市': city_2,
    '所属行政区': area_2,
    '所在街道': item.find('.positionInfo a').eq(-1).text(),
    '小区名称': item.find('.positionInfo a').eq(0).text(),
    '价格': item.find('.totalPrice').text().replace('万',
'0000').replace('参考价: ', '').replace('.', '').replace(
' ', ''),
    '建筑面积': mianzhi,

```

```

        '每平方单价': item.find('.unitPrice span').text().replace(',', ''),
        '户型': huxing,
        '装修情况': zhuangxiu,
        '房屋朝向': chaoxiang,
        '楼层': loucheng,
        '楼龄': louling,
        '建筑类型': jianzhuleixing,
        '发布时间': item.find('.followInfo').text().replace(' ',
    '').split('/')[0].replace('发布', ''),
        '关注人数': item.find('.followInfo').text().replace(' ',
    '').split('/')[0],
        '房源标签': item.find('.tag span').text() or '暂无数据',
        # 作为去重和表连接的字段
        '房源链接': item.find('.title a').attr('href'),
        # 页面的src属性并非真实的图片url, 真正的url在属性data-original里
        '房源图片链接': item.find('.noresultRecommend .lj-lazy').attr('data-
original'),
        '当前信息时间': datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S'),
        '是否已获取该房源明细信息': '0'

    }

    if house_info['小区名称'] != '' and chewei != '车位':
        print(house_info)
        save_to_mysql('链家全国二手房基本信息表', house_info)

# selenium方式不容易引起反爬
def browser(province_3, city_3, area_3, link_2):
    try:
        bro.get(link_2)
        try:
            bro.find_element(By.XPATH, '//*[@id="content"]/div[1]/div[2]/h2/span')
        except NoSuchElementException:
            print('=====未能找到指定元素, 可能遇到爬虫
            =====')
            sys.exit()
        response = bro.page_source
        analysis(province_3, city_3, area_3, response)
    except TimeoutException:
        print('=====连接超时, 请检查网络, 30秒后重试
        =====')
        bro.quit()
        sys.exit()

def load_from_mysql():
    cursor = DB.cursor()
    # 通过SQL语句获取指定区域的信息
    sql = " select 省份,城市,区域,街道,街道主键,街道链接,当前街道房源数量,当前街道房源页数
from 房源数量中间表 WHERE 当前街道房源是否全部已获取 = 0 " \
        "AND 当前街道房源数量 > 0"
    # 只取未获取的街道来执行
    cursor.execute(sql)
    info = cursor.fetchall()
    DB.commit()
    for i in info:

```

```

        print('===== ' + i[0] + '省（直辖市）' + i[1] + '市'
              + i[2] + '(县)下属' + i[3] + '共找到' + i[
                6] + '套二手房源=====\\n')
        for u in range(1, int(i[7]) + 1):
            # request 速度快, 容易反爬
            request(i[0], i[1], i[2], i[5] + 'pg' + str(u))
            # browser 速度慢, 不容易反爬
            # browser(i[0], i[1], i[2], i[5] + 'pg' + str(u))
            # 一个城市所有页翻完才能去标记
            mark(i[4])
        cursor.close()

def mark(symbol):
    cursor = DB.cursor()
    sql_update = 'UPDATE 房源数量中间表 SET 当前街道房源是否全部已获取 = %s WHERE 街道主
键 = %s'
    try:
        cursor.execute(sql_update, (1, symbol))
        DB.commit()
    except pymysql.Error as e:
        DB.rollback()
    cursor.close()

# 存储到MySQL数据库, 建好表
def save_to_mysql(key, result):
    cursor = DB.cursor()
    table = key
    keys = ','.join(result.keys())
    values = ','.join(['%s' * len(result)])

    sql = 'INSERT INTO {table}({keys}) VALUES ({values}) ON DUPLICATE KEY
UPDATE'.format(table=table, keys=keys,

                values=values)
    update = ','.join([" {key} = %s".format(key=key) for key in result])
    sql += update
    try:
        if cursor.execute(sql, tuple(result.values()) * 2):
            print('存储到MySQL数据库成功')
            DB.commit()
    except pymysql.Error as e:
        DB.rollback()
        print("Mysql Error %d: %s" % (e.args[0], e.args[1]))

# 驱动函数
def main():
    load_from_mysql()

if __name__ == '__main__':
    # 设置浏览器参数对象
    option = ChromeOptions()
    # 关闭chrome弹出得密码保存框和网页通知询问
    prefs = {"": "", "credentials_enable_service": False,
            "profile.password_manager_enabled": False,

```

```
        "profile.default_content_setting_values": {'notifications': 2}}
option.add_experimental_option("prefs", prefs)
# 关闭自动化测试提醒
option.add_experimental_option("excludeSwitches", ['enable-automation'])
# 屏蔽webdriver特征
option.add_argument("--disable-blink-features")
option.add_argument("--disable-blink-features-AutomationControlled")
# TODO 使用无头 提高速度
option.add_argument('--headless')
bro = webdriver.Chrome(options=option)
# 隐藏selenium特性
bro.execute_cdp_cmd("Page.addScriptToEvaluateOnNewDocument", {
    "source": """"Object.defineProperty(navigator, 'webdriver', {get: () =>
undefined})""",
})
print('=====开始爬取
=====')

# 初始化MySQL
try:
    DB = pymysql.connect(host='localhost', port=3307, user='root',
password='000000', database='house')
except pymysql.Error as e:
    print("Mysql Error %d: %s" % (e.args[0], e.args[1]))

main()
sleep(5)
print('=====爬取完成, 释放资源
=====')

bro.quit()
DB.close()
```

效果图：

已获取房源明细信息: '0'																
存储到MySQL数据库成功																
{ '房源主键': 'zs105104808695', '省份': '广东', '城市': '中山', '所属行政区': '港口镇', '所在街道': '港口市场', '小区名称': '上乘世纪公园', '价格': '930000', '建筑面积': '63平米', '每平方单价': '14762元/平', '户型': '1室1厅', '装修情况': '精装', '房屋朝向': '南', '楼层': '中楼层(共25层)', '楼龄': '2019年建', '建筑类型': '板楼', '发布时间': '一年前', '关注人数': '2人关注', '房源标签': 'VR看装修 随时看房', '房源链接': 'https://zs.lianjia.com/ershoufang/105104808695.html', '房源图片链接': 'https://image1.ljcdn.com/110000-inspection/pc1_0f18upM2K_1.jpg.296x216.jpg', '当前信息时间': '2022-03-01 11:32:06', '是否已获取该房源明细信息': '0' }																
存储到MySQL数据库成功																
房源主键	省份	城市	所属行政区	所在街道	小区名称	价格	建筑面积	每平方单价	户型	装修情况	房屋朝向	楼层	楼龄	建筑类型	发布时间	关注人数
aq103103160963	安徽	安庆	宜秀区	大桥	旺旺家缘	1200000	176平米	6819元/平	3室2厅	简装	南北	高楼层(共7层)	暂无数据	板楼	一年前	0人关注
aq103103161052	安徽	安庆	宜秀区	大桥	永安公馆	1500000	127平米	11812元/平	3室2厅	毛坯	南北	中楼层(共24层)	暂无数据	板塔结合	一年前	3人关注
aq103103161202	安徽	安庆	迎江区	德宽路	行管局宿舍	5850000	78平米	7500元/平	2室2厅	精装	南北	中楼层(共7层)	暂无数据	板楼	一年前	2人关注
aq103103161217	安徽	安庆	宜秀区	大桥	置地百悦府	1150000	118平米	9746元/平	3室2厅	毛坯	南	中楼层(共34层)	暂无数据	板楼	一年前	2人关注
aq103103161287	安徽	安庆	宜秀区	大桥	凯旋尊邸	3600000	406.45平米	8858元/平	6室3厅	毛坯	南北	3层	板楼	双拼别墅	一年前	13人关注
aq103103161298	安徽	安庆	宜秀区	大桥	凯旋尊邸	1600000	130平米	12308元/平	3室2厅	精装	南北	31层	暂无数据	板楼	一年前	1人关注
aq103103161365	安徽	安庆	宜秀区	大桥	嘉禾园	1000000	127平米	7875元/平	3室2厅	毛坯	南	低楼层(共16层)	暂无数据	板楼	一年前	3人关注
aq103103161602	安徽	安庆	宜秀区	大桥	恒禾东尚	1150000	124平米	9275元/平	3室2厅	毛坯	南	低楼层(共18层)	暂无数据	板楼	一年前	2人关注
aq103103162735	安徽	安庆	宜秀区	大桥	恒禾东尚	990000	106平米	9340元/平	3室2厅	毛坯	南	高楼层(共18层)	暂无数据	板楼	一年前	0人关注
aq103103162764	安徽	安庆	宜秀区	大桥	博雅花苑	1100000	102.18平米	10766元/平	3室2厅	简装	南北	中楼层(共9层)	暂无数据	板楼	一年前	0人关注
aq103103175561	安徽	安庆	宜秀区	大桥	光彩商务花园	2380000	235平米	10128元/平	7室3厅	精装	南北	3层	板楼	联排别墅	一年前	16人关注
aq103103189652	安徽	安庆	宜秀区	大桥	永安公馆	10980000	125平米	8784元/平	3室2厅	毛坯	南北	低楼层(共18层)	暂无数据	板塔结合	一年前	11人关注
aq103103224745	安徽	安庆	宜秀区	大桥	书香一品南园	1100000	121平米	9091元/平	3室2厅	毛坯	南	中楼层(共18层)	暂无数据	板楼	一年前	2人关注
aq103103288869	安徽	安庆	迎江区	德宽路	青年新村	700000	71.39平米	9806元/平	3室1厅	简装	南	高楼层(共11层)	暂无数据	板塔结合	一年前	0人关注
aq103103293795	安徽	安庆	宜秀区	大桥	文苑世家	1350000	135平米	10000元/平	3室2厅	毛坯	南	中楼层(共11层)	暂无数据	板楼	一年前	0人关注
aq103103367802	安徽	安庆	宜秀区	大桥	嘉禾园	1200000	127平米	9449元/平	3室2厅	毛坯	南北	中楼层(共16层)	暂无数据	板楼	一年前	0人关注
aq103103367852	安徽	安庆	宜秀区	大桥	迎春颐和城	1650000	206.33平米	7997元/平	5室2厅	毛坯	南北	高楼层(共18层)	暂无数据	板楼	一年前	1人关注
aq103103367859	安徽	安庆	宜秀区	大桥	迎春颐和城	1490000	186.8平米	7977元/平	5室3厅	毛坯	南北	高楼层(共18层)	暂无数据	板楼	一年前	8人关注
aq103103454758	安徽	安庆	宜秀区	大桥	凯旋尊邸	1480000	183平米	8088元/平	4室2厅	毛坯	南北	29层	暂无数据	板楼	一年前	2人关注

整体的数据体量达到330万+了，还没有完全爬完，预估是能破360万+，这个数据是经过去重，去广告，去车位信息的了。

zs105109896265	广东	中山	阜沙镇	阜沙镇	大盛花园
zs105109897036	广东	中山	东凤镇	东凤镇	逸湖半岛
zs105109897249	广东	中山	南区	广珠公路南	金水湾
zs105109897268	广东	中山	石岐区	第一城小学	正和中州
zs105109897461	广东	中山	火炬	博览中心	城东名门
zs105109897623	广东	中山	港口镇	港口市场	富元港景峰

```

< 
+ - ✓ ✕ C ■
SELECT * FROM `house`.`链家全国二手房基本信息表` LIMIT 3304000,1000

```

后续也许还有朋友需要获取每套房源下面的具体信息：譬如经纪公司，小区特色，反正房源链接下面的所有信息，我也写了代码，但是还没有改造PyMySQL方法和设计表，需要的可以自己改造下，我直接放出代码。

```

import random
from time import sleep
import pymongo
# 导入pyquery解析页面
import requests
from concurrent.futures import ThreadPoolExecutor
import pymysql
from pyquery import PyQuery as pq
from selenium import webdriver
# 让selenium规避被检测风险
from selenium.webdriver import ChromeOptions, ActionChains
# 导入显示等待参数
from selenium.webdriver.common.by import By
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.support.ui import WebDriverWait

def load_from_mongodb():
    # 从Mongodb中取值（详情页网址）
    for item in db['广州市各街道二手房源基本信息表'].find({},
                                                              {'_id': 0, '房源链接': 1}):
        try:
            get_details(item['房源链接'])
            print('=====开始获取下一条明细')
            print('=====')
        except:
            print('=====出现反爬虫')
            print('=====')

def load_from_mysql():
    a = 0
    cursor = DB.cursor() # 使用cursor()方法获取操作游标
    sql = "select 房源链接 from " + 'house_guangzhou_list_etl'
    cursor.execute(sql)
    info = cursor.fetchall()
    DB.commit()
    for i in info:

```

```

        try:
            print('=====开始获取下一条明细=====')
            get_details(i['房源链接'])
            a += 1
            print('这是第' + str(a) + '条')
            if a % 1000 == 0:
                print('一千条了，休息一下')
                sleep(600)
        except:
            print('=====反爬=====')

    cursor.close()

def get_details(link):
    headers = {

        'User-Agent': 'Mozilla/5.0 (windows NT 10.0; win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.102 Safari/537.36
Edg/98.0.1108.55',
        'Referer': 'https://lianjia.com/',

    }
    # bro.get(link)
    # html = bro.page_source
    response = requests.get(url=link, headers=headers).text
    doc = pq(response)
    house_details = {
        # 房源固定信息
        '小区名称': doc.find('.communityName .info').text(),
        '价格': (doc.find('.price-container .total').text() + doc.find('.price-
container .unit').
            text()).replace('万', '0000').replace('.', ''),
        '每平方单价': doc.find('.text .unitPrice span').text(),
        '所属行政区': doc.find('.areaName .info a').eq(0).text(),
        '所在街道': doc.find('.areaName .info a').eq(1).text(),
        '房子ID': doc.find('.houseRecord .info').text().replace('举报', ''),
        '建筑时间': doc.find('.houseInfo .area .subInfo').text().split('/')
[0].replace('建', ''),

        # 房源固定信息
        '房屋户型': doc.find('.base .content li').eq(0).text().replace('房屋户型',
''),
        '所在楼层': doc.find('.base .content li').eq(1).text().replace('所在楼层',
''),
        '建筑面积': doc.find('.base .content li').eq(2).text().replace('建筑面积',
''),
        '户型结构': doc.find('.base .content li').eq(3).text().replace('户型结构',
''),
        '套内面积': doc.find('.base .content li').eq(4).text().replace('套内面积',
''),
        '建筑类型': doc.find('.base .content li').eq(5).text().replace('建筑类型',
''),
        '房屋朝向': doc.find('.base .content li').eq(6).text().replace('房屋朝向',
''),
        '建筑结构': doc.find('.base .content li').eq(7).text().replace('建筑结构',
''),
    }

```

```

        '装修情况': doc.find('.base .content li').eq(8).text().replace('装修情况',
    ''),
        '梯户比例': doc.find('.base .content li').eq(9).text().replace('梯户比例',
    ''),
        '配备电梯': doc.find('.base .content li').eq(10).text().replace('配备电梯',
    ''),

    # 房源交易信息
    '挂牌时间': doc.find('.transaction .content
li').eq(0).find('span').eq(-1).text(),
    '交易权属': doc.find('.transaction .content
li').eq(1).find('span').eq(-1).text(),
    '上次交易': doc.find('.transaction .content
li').eq(2).find('span').eq(-1).text(),
    '房屋用途': doc.find('.transaction .content
li').eq(3).find('span').eq(-1).text(),
    '房屋年限': doc.find('.transaction .content
li').eq(4).find('span').eq(-1).text(),
    '产权所属': doc.find('.transaction .content
li').eq(5).find('span').eq(-1).text(),
    '抵押信息': doc.find('.transaction .content
li').eq(6).find('span').eq(-1).text(),
    '房本备件': doc.find('.transaction .content
li').eq(7).find('span').eq(-1).text(),

    # 小区信息
    '核心卖点': doc.find('.introContent
.baseattribute').eq(0).find('div').eq(1).text().replace('\n', ':'),
    '小区介绍': doc.find('.introContent
.baseattribute').eq(1).find('div').eq(1).text().replace('\n', ':'),
    '周边配套': doc.find('.introContent
.baseattribute').eq(2).find('div').eq(1).text().replace('\n', ':'),
    '交通出行': doc.find('.introContent
.baseattribute').eq(3).find('div').eq(1).text().replace('\n', ':'),

    # 经纪人信息
    '经纪人': doc.find('.ke-agent-sj-top .ke-agent-sj-fr .ke-agent-sj-
name').text(),
    '经纪公司': doc.find('.ke-agent-sj-top .ke-agent-sj-fr .ke-agent-sj-
tag').text(),
    '经纪电话': doc.find('.ke-agent-sj-container .ke-agent-sj-sdk-f-0 .ke-
agent-sj-phone').text().replace(' ',

        ''),

    # 链接
    '房屋照片': doc.find('.img .imgContainer img').attr('src'),
    '房源连接': link
    }
    print(house_details)
    save_to_mongo(house_details)

# 设置请求头

# 写入到mongodb
def save_to_mongo(result):

```

```

try:
    # 新版本pymongo弃用了.insert方法改用.insert_one
    if db[MONGO_TABLE].insert_one(result):
        print('存储到MongoDB数据库成功')
except Exception:
    print('存储到MongoDB数据库失败' + result)

# 驱动函数
def main():
    load_from_mysql()

if __name__ == '__main__':
    # 初始化MongoDB
    MONGO_URL = 'localhost'
    MONGO_DB = 'house'
    client = pymongo.MongoClient(MONGO_URL)
    db = client[MONGO_DB]

    # 初始化MySQL
    DB = pymysql.connect(host='localhost', port=3307, user='root',
        password='000000', database='house',
        charset='utf8mb4',
        # 把查询的返回集以字典形式保存
        cursorclass=pymysql.cursors.DictCursor)

    MONGO_TABLE = '广州市各街道二手房源明细信息表1'

    # 设置浏览器参数对象
    option = ChromeOptions()
    # 关闭chrome弹出得密码保存框和网页通知询问
    prefs = {"": "", "credentials_enable_service": False,
        "profile.password_manager_enabled": False,
        "profile.default_content_setting_values": {'notifications': 2}}
    option.add_experimental_option("prefs", prefs)
    # 关闭自动化测试提醒
    option.add_experimental_option("excludeSwitches", ['enable-automation'])
    # 屏蔽webdriver特征
    option.add_argument("--disable-blink-features")
    option.add_argument("--disable-blink-features-AutomationControlled")
    # TODO 使用无头
    option.add_argument('--headless')
    # 初始化引擎参数
    # bro = webdriver.Chrome(options=option)
    # 反淘宝js检测手段 解决滑块验证码
    # bro.execute_cdp_cmd("Page.addScriptToEvaluateOnNewDocument", {
    #     "source": """"Object.defineProperty(navigator, 'webdriver', {get: () =>
    undefined})""",
    # })

    # 配置显示等待参数传入引擎对象和最大等待时间
    # wait = WebDriverWait(bro, 60)
    main()
    print('=====当前列表房源明细全部爬取完成
    =====')
    sleep(5)
    # bro.quit()

```


最后放出三个py文件，分别对应着全部中间表获取，基本信息爬取，详细信息爬取。

也把所有SQL文件直接放出给需要直接做分析清洗的人

可以加入邮件预警系统，解决爬虫中断问题，链家本身反爬不严重，但只是学习用途没有开启多线程和代理池，速度也已经可以了，不要做的太过分！

时间：2022-02-26T01:24:07.198Z

网址：<https://cq.lianjia.com/ershoufang/laodonglu/>

封禁原因

访问频次过高或周边同事/邻居对贝壳房源存在恶意爬取行为