[智干商业智能部，关于美国亚马逊电子商务平台商品排名数据相关需求，全网爬虫设计，开发和使用文档]

[亚马逊全网爬虫文档]

[时间:2016/10/20 汇编：商业智能部]

[广州智干电子商务有限公司]

**目录**

[一．前言 1](#_Toc24913)

[1. 需求 1](#_Toc1277)

[2. 开发安排 1](#_Toc29060)

[二．亚马逊美国站爬虫软件设计 2](#_Toc7801)

[1. 类目 2](#_Toc5226)

[2. 存储（空间） 3](#_Toc32468)

[3. 速度（时间） 4](#_Toc21199)

[4. 局限（反爬） 4](#_Toc30055)

[5. 分布式 5](#_Toc27001)

[三．亚马逊美国站爬虫软件使用 6](#_Toc4961)

[1. 修改配置文件 7](#_Toc30266)

[1.1 全局配置 7](#_Toc4913)

[1.2 日志配置 9](#_Toc30268)

[2. 建立数据库 9](#_Toc16377)

[2.1 基础数据库smart\_base 9](#_Toc26646)

[2.2 商品数据库smart\_item 10](#_Toc1523)

[3. 爬虫使用bin/spider 11](#_Toc14128)

[3.1 urlspider排名类目爬虫 11](#_Toc23218)

[3.2 ratespider排名爬虫 13](#_Toc879)

[4. 辅助工具使用bin/tool 14](#_Toc25756)

[4.1 proxyfiletool.py 14](#_Toc12117)

[4.2 proxymysqltool.py 14](#_Toc17178)

[4.3 createtabletool.py 15](#_Toc12269)

[4.4 urlspidertool.py 15](#_Toc9541)

[4.5 validurlspidertool.py 16](#_Toc13621)

[4.6 testproxytool.py 17](#_Toc2856)

[5. 客户端工具 17](#_Toc14072)

[5.1 client/exportdata.py 导数据 17](#_Toc11714)

[5.2 client/saveproxyredis.py 拯救IP 17](#_Toc4387)

[6. 外部文件数据存储架构 17](#_Toc11378)

[四．服务器规划 19](#_Toc29727)

[五．服务器行为准则 20](#_Toc21111)

[1. 基本命令 20](#_Toc20673)

[2. 规范 20](#_Toc7585)

[3. 维护 20](#_Toc25787)

[六．英文介绍 23](#_Toc14189)

# 一．前言

智干电子商务有限公司主要从事境外电商运营，因为业务需要，需要建立商业智能部门。目前的需求是实现网络爬虫抓取电商平台数据，进行亚马逊商品数字运营。

## 1. 需求

需求如下：

1. 实现爬虫抓取亚马逊各级类目的最小类目Top100的排名，保存在数据库，方便抽取成EXCEL文件给业务人员使用,

要求爬虫稳定，爬取速度适宜，不影响日常使用。团队使用Python语言开发爬虫，替代火车头第三方软件，以后会进行优化，使用Golang或Java开发。爬取美国站优先。

3. 开发亚马逊小工具，如亚马逊后台的各种数据抓取工具

4. 开发其他爬虫，如京东爬虫

5. 对产生的数据进行分析，挖掘有用的信息

## 开发安排

爬取美国站商品分类下排名数据，进入商品详情页获取大类排名。开发使用python，轮转IP，轮转Useragent，支持并行抓取，数据保存在数据库mysql，考虑速度问题，分模块，分类目多只爬虫分布式爬取。数据库设计为该项目较重要部分。

任务1：IP池构造，UA池构造，基础代码架构

任务2：开发类目爬虫抓取所有类目，提取URL，存入数据库（需设计）

任务3：开发商品排名爬虫，提取URL链接，存入数据库（需设计）

任务4：开发商品抓取分布模块，批量抓取商品详情页，存入数据库(需设计)  
任务5：爬虫测试,部署生产环境，把爬虫部署至生产环节

任务6：撰写设计和使用文档

时间安排：两周

# 二．亚马逊美国站爬虫软件设计

## 1. 类目

通过爬取亚马逊类目，获得（只统计到五级类目，后来增加到六级，未计入计算）

| **大类名** | **类目数** | **最小类目数** |
| --- | --- | --- |
| Arts\_ Crafts & Sewing | 638 | 535 |
| Automotive | 2900 | 2538 |
| Baby | 336 | 274 |
| Beauty | 358 | 298 |
| Camera & Photo | 237 | 191 |
| Cell Phones & Accessories | 66 | 54 |
| Clothing | 365 | 298 |
| Computers & Accessories | 297 | 249 |
| Electronics | 1829 | 1510 |
| Health & Personal Care | 1054 | 875 |
| Home & Kitchen | 2016 | 1685 |
| Home Improvement | 1484 | 1243 |
| Industrial & Scientific | 3230 | 2752 |
| Jewelry | 176 | 143 |
| Kitchen & Dining | 935 | 804 |
| Musical Instruments | 592 | 473 |
| Office Products | 837 | 705 |
| Patio\_ Lawn & Garden | 551 | 459 |
| Pet Supplies | 488 | 393 |
| Shoes | 200 | 170 |
| Sports & Outdoors | 1688 | 1468 |
| Toys & Games | 746 | 614 |
| Video Games | 722 | 564 |
| Watches | 22 | 17 |
| 总计 | 21767 | 18312 |

其中有些类目出现重复，如归属到同一个父类，个数大概是一千多个，由于大类排名唯一，所以只抓取一个。

实际抓取类目整理如下：



## 2. 存储（空间）

总共18312个最小类目，每个类目100件商品，根据抓取的存储来看，每个类目一天占用4K的数据库存储空间，本地文件占用92M，计算所得，一天抓取的数据超过180万，占用数据库空间73M，占用本地存储空间1.68T（本地文件要定时清理）

计算所得：

| 时长 | 数据库数据量 | 本地数据量 |
| --- | --- | --- |
| 1天 | 73M | 1.68T |
| 一个月 | 2.19G | 50.4T |
| 一年 | 26.645G | 613.2T |

所以存储空间绰绰有余。

## 3. 速度（时间）

采用多台机器分配任务，每台机器并行进程任务，缩小时间。

在单台机器上并行10个进程，抓取11个类目的时间是（带宽一般家庭）：198秒（更快可弥补暂停时间，则代理IP问题解决较易）

那么一台机器开十个进程，抓取所有类目抓取需要91.6个小时，软件开发后（假设带宽能够支撑并行数）

| 机器数 | 并行数 | 时间 |
| --- | --- | --- |
| 1 | 1 | 915.6小时 |
| 1 | 10 | 91.56小时 |
| 1 | 20 | 45.78小时 |
| 2 | 10 | 45.78小时 |
| 2 | 20 | 22.89小时 |
| 5 | 10 | 18.31小时 |
| 5 | 20 | 9.15小时 |
| 10 | 20 | 4.575小时 |
| 20 | 20 | 2.28小时 |

## 4. 局限（反爬）

代理IP数量不够时，并行程序轮询IP时会大概率同时使用同个IP，导致机器人限制，IP随机数选取需要设计。

我们假设在同一时间，一条IP需暂停3秒才符合人类操作，不会被反爬，那么我们计算所得，IP数必然需要大于并行数，且需暂停0-3秒，不暂停时，IP有概率冲突，计算需数学建模（排队论）。

| 机器数 | 并行数 | 时间 | IP数最少（时间翻倍） |
| --- | --- | --- | --- |
| 1 | 1 | 915.6小时 | 1 |
| 1 | 10 | 91.56小时 | 10 |
| 1 | 20 | 45.78小时 | 20 |
| 2 | 20 | 22.89小时 | 40 |
| 5 | 10 | 18.31小时 | 50 |
| 5 | 20 | 9.15小时 | 100 |
| 10 | 20 | 4.575小时 | 200 |
| 20 | 20 | 2.28小时 | 400 |

所以如何使时间更短，IP有效利用，我们将IP打到redis形成IP池消息队列，并行程序从redis pop IP，根据ip时间确定要暂停还是继续，如果没有IP阻塞，取到IP使用完后打回消息队列里，当IP被反时，可配置打到另外一机器人队列，开另一程序监控机器人消息队列，实现人工打码（已经实现）。确定一个IP使用间隔在一个时间周期里，不出现冲突，冲突会导致反爬。

## 5. 分布式

由于数据库设计，数据库名人为指定，表名为数据库ID，总共有一万多张表，数据库待改用mongodb，且有一部分抓取过程可用redis仿照分布式ip池，实现动态增加机器而不用手工改参数。（待实现）

# 三．亚马逊美国站爬虫软件使用

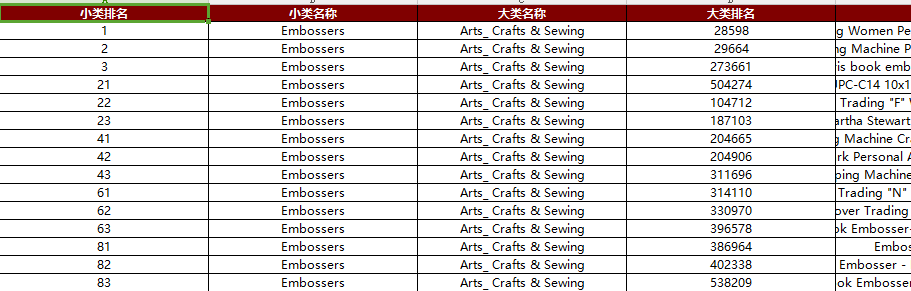
爬虫有两只

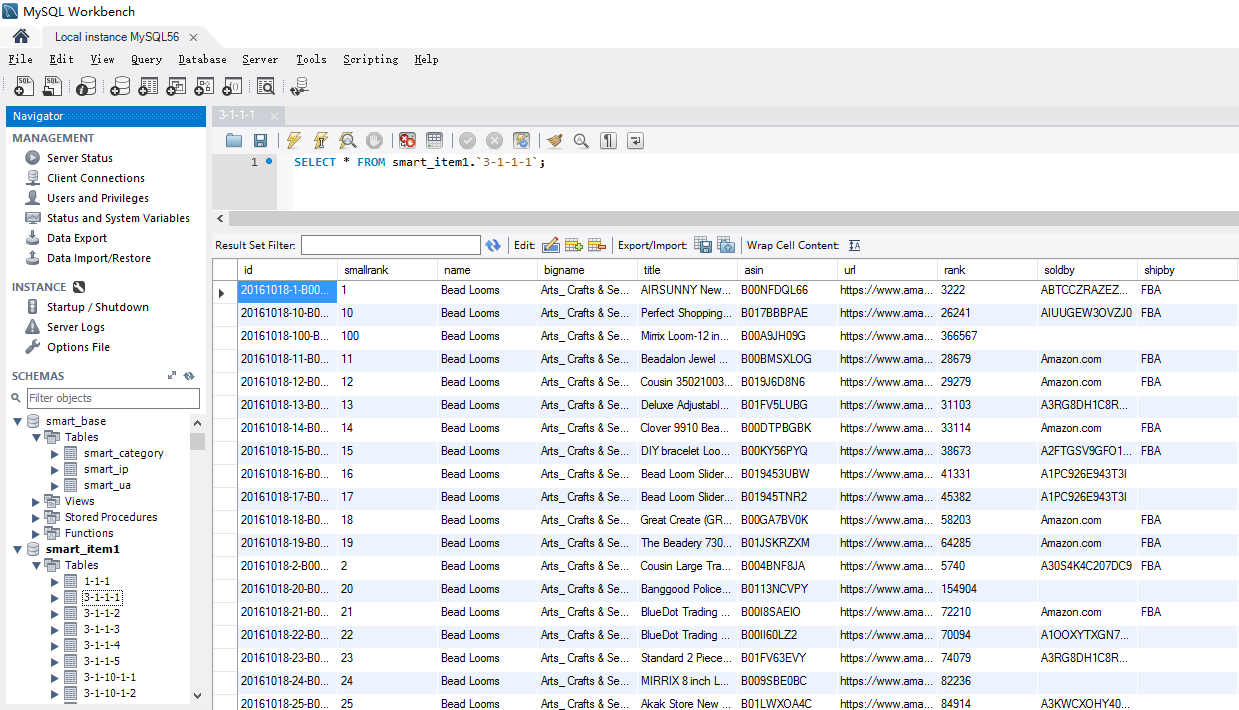
第一只是亚马逊商品类目爬虫，总共2万多个类目，保存在一个文件夹中，并需要使用辅助函数处理存入数据库。

第二只是亚马逊排名数据爬虫，支持并行，分布式，高容错，反爬虫等功能。

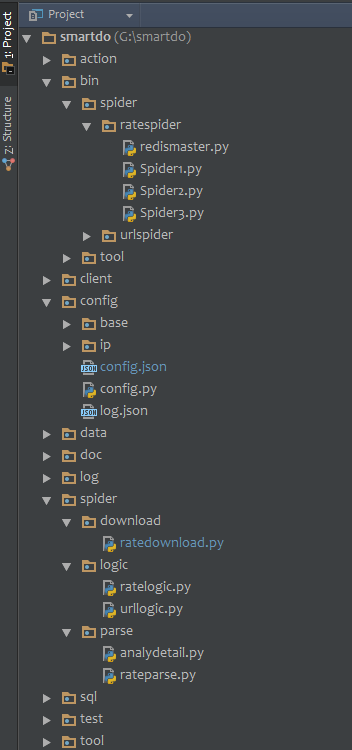
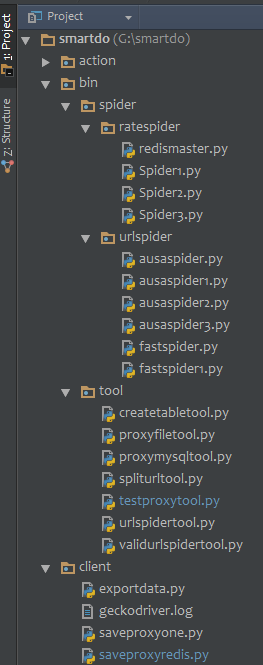
爬虫使用需要先更改配置文件，创建数据库，运行执行文件

文件结构和数据结果如下图：





代码结构如下：

## 修改配置文件

全局配置config/config.json和日志配置log.json （#连同后面内容需去掉）

### 1.1 全局配置

{

"company": "Smartdo Co.,Ltd",

"version": "v1",

"developer": "陶彦百，陈锦瀚",

"time": "2016-11",

"datadir":"/data/db/usa", # 重要数据保存位置，全路径

"manyua":true, # 多浏览器支持

"manycookie":true, # cookie支持

"limitip":false, # 限制IP，根据iperror

"mysqlipnum":2000,# 限制从数据库读到的IP数

"iperror":20000,# 代理IP的反爬最大数量（存于数据库，ip选择从数据库select后判断）

"itemnum":60,# 商品类目抓取的最大置信数，大于该数，不重抓

"sleeptimes":10, # 非redis ip池爬虫下载睡眠时间秒

"koip":true,# 遭遇机器人限制时是否剔除该IP，（redis池踢到一队列，其他方式直接踢掉后写数据库）

"localkeep":false,# 详情页是否保存本地（很大，建议关闭）

"catchurl":["1","2","3"],# 根据catchbywhich字段，select其下的子类

"catchbywhich":"database", # select的字段名，可选database或者bigpname

"processnum":50,# 并行进程数

"urlnum":20000,# 类目数限制（建议20000，不限制）

"basedb": {# 代理IP和类目信息数据库位置

"host": "45.41.88.189",

"user": "root",

"pwd": "smart2016",

"db": "smart\_base"

},

"ipinmysql":true,# 从数据库拿ip

"redispool":true, # 分布式ip池开启（需安装redis)

"redispoolname":"ippool",# ip池名称

"redispoolfuckname":"ippoolfuck",# 机器人池名称

"redispoolnumber":3, # 同时开启的机器数，根据机器数构造若干个池

"redispoolsleeptimes":10,# ip暂停时间

"rediserrmaxtimes":2,# 机器人容忍数量

"redispoolconfig":{"host":"45.41.88.189","pwd":"smart2016","port":6379},# redis配置

"dbprefix":"db",# 类目数据库字段的前缀，如1，拼接后db1，找到下面真实的数据库地址

"db1": {"host": "45.41.88.189", "user": "root", "pwd": "smart2016", "db": "smartdb1"},

"db2": {"host": "45.41.88.189", "user": "root", "pwd": "smart2016", "db": "smartdb2"},

"db3": {"host": "45.41.88.189", "user": "root", "pwd": "smart2016", "db": "smartdb3"},

"db4": {"host": "45.41.88.189", "user": "root", "pwd": "smart2016", "db": "smartdb4"},

"db5": {"host": "45.41.88.189", "user": "root", "pwd": "smart2016", "db": "smartdb5"},

"db6": {"host": "45.41.88.189", "user": "root", "pwd": "smart2016", "db": "smartdb6"},

"db7": {"host": "45.41.88.189", "user": "root", "pwd": "smart2016", "db": "smartdb7"},

"db8": {"host": "45.41.88.188", "user": "root", "pwd": "smart2016", "db": "smartdb8"},

"db9": {"host": "45.41.88.188", "user": "root", "pwd": "smart2016", "db": "smartdb9"},

"db10": {"host": "45.41.88.188", "user": "root", "pwd": "smart2016", "db": "smartdb10"},

"db11": {"host": "45.41.88.188", "user": "root", "pwd": "smart2016", "db": "smartdb11"},

"db12": {"host": "45.41.88.188", "user": "root", "pwd": "smart2016", "db": "smartdb12"},

"db13": {"host": "45.41.88.188", "user": "root", "pwd": "smart2016", "db": "smartdb13"},

"db14": {"host": "45.41.88.188", "user": "root", "pwd": "smart2016", "db": "smartdb14"},

"db15": {"host": "45.41.88.187", "user": "root", "pwd": "smart2016", "db": "smartdb15"},

"db16": {"host": "45.41.88.187", "user": "root", "pwd": "smart2016", "db": "smartdb16"},

"db17": {"host": "45.41.88.187", "user": "root", "pwd": "smart2016", "db": "smartdb17"},

"db18": {"host": "45.41.88.187", "user": "root", "pwd": "smart2016", "db": "smartdb18"},

"db19": {"host": "45.41.88.187", "user": "root", "pwd": "smart2016", "db": "smartdb19"},

"db20": {"host": "45.41.88.187", "user": "root", "pwd": "smart2016", "db": "smartdb20"},

"db21": {"host": "45.41.88.187", "user": "root", "pwd": "smart2016", "db": "smartdb21"},

"db22": {"host": "45.41.88.187", "user": "root", "pwd": "smart2016", "db": "smartdb22"}

}

### 1.2 日志配置

"root": {

"level": "ERROR", # 可改CRITICAL > ERROR > WARNING > INFO > DEBUG > NOTSET 建议不改

"handlers": [

"console",

"error\_file\_handler"

]

}

日志将自动以时间20161018创建文件夹，20161018-小时为文件名来存储，存储于log文件夹下

## 2. 建立数据库

### 2.1 基础数据库smart\_base

CREATE TABLE `smart\_category` (

`id` varchar(100) NOT NULL,

`url` varchar(255) DEFAULT NULL COMMENT '类目链接',

`name` varchar(255) DEFAULT NULL COMMENT '类目名字',

`level` tinyint(4) DEFAULT NULL COMMENT '类目级别',

`pid` varchar(100) DEFAULT NULL COMMENT '父类id',

`createtime` datetime DEFAULT NULL COMMENT '创建时间',

`updatetime` datetime DEFAULT NULL COMMENT '更新时间',

`isvalid` tinyint(4) DEFAULT '0' COMMENT '是否有效',

`page` tinyint(4) DEFAULT '5' COMMENT '抓取页数',

`database` varchar(255) DEFAULT NULL COMMENT '存储数据库',

`col1` varchar(255) DEFAULT NULL COMMENT '预留字段',

`col2` varchar(255) DEFAULT NULL,

`col3` varchar(255) DEFAULT NULL,

`bigpname` varchar(255) DEFAULT NULL COMMENT '大类名字',

`bigpid` varchar(100) DEFAULT NULL COMMENT '大类ID',

`ismall` tinyint(4) DEFAULT '0' COMMENT '是否最小类',

PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='类目';

id为标志位，如1，1-1，1-1-1，表明是第一个分类下面的第一个分类下面的第一个分类。

爬虫原理是根据bigpname筛选出isvalid为1的类目，然后根据进程数平均分配，开始抓取，抓取之前判断database字段在配置文件

是否存在，存在的话判断该数据库是否存在这张表，表名为id，没有则报错退出。

CREATE TABLE `smart\_ip` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`ip` varchar(45) NOT NULL,

`createtime` datetime DEFAULT NULL COMMENT '添加时间',

`updatetime` datetime DEFAULT NULL COMMENT '更新时间',

`failtimes` int(11) DEFAULT '0' COMMENT '失效次数',

`zone` varchar(200) DEFAULT NULL COMMENT '区域',

`col1` varchar(200) DEFAULT NULL COMMENT '预留字段',

`col2` varchar(200) DEFAULT NULL,

`col3` varchar(200) DEFAULT NULL,

PRIMARY KEY (`id`),

UNIQUE KEY `ip\_UNIQUE` (`ip`)

) ENGINE=InnoDB AUTO\_INCREMENT=9218 DEFAULT CHARSET=utf8 COMMENT='IP池';

代理IP数据表zone字段指明地理位置，如美国，failtimes指明失效次数，如果大于配置文件所需次数，抽取时被弃用。

### 2.2 商品数据库smart\_item

CREATE TABLE `smart\_item`.`1-1-1` (

`id` VARCHAR(255),

`smallrank` INT NULL COMMENT '小类排名',

`name` VARCHAR(255) NULL COMMENT '小类名',

`bigname` VARCHAR(255) NULL COMMENT '大类名',

`title` TINYTEXT NULL COMMENT '商品标题',

`asin` VARCHAR(255) NULL,

`url` VARCHAR(255) NULL,

`rank` INT NULL COMMENT '大类排名',

`soldby` VARCHAR(255) NULL COMMENT '卖家',

`shipby` VARCHAR(255) NULL COMMENT '物流',

`price` FLOAT NULL COMMENT '价格',

`score` FLOAT NULL COMMENT '打分',

`commentnum` INT NULL COMMENT '评论数',

`commenttime` VARCHAR(255) NULL COMMENT '第一条评论时间',

`createtime` DATETIME NULL,

PRIMARY KEY (`id`) )ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='类目表';

表名为类目id，id字段命名为20161018-1-B00NFDQL66 时间-小类排名-asin

类目数据要放在哪个数据库，需要手动建表（我写了python脚本），先在smart\_category表将该类目isvalid置为1且database字段填入ratedb1，如下

"ratedb1": { # 类目抓取数据存储的数据库，类目信息database字段指明了该数据库

"host": "192.168.0.152",

"user": "bai",

"pwd": "123456",

"db": "smart\_item1"

}

然后配置文件真正的数据库是smart\_item1，在该库建立1-1-1表（以此类推）

## 3. 爬虫使用bin/spider

### 3.1 urlspider排名类目爬虫

（可不使用，直接导入数据库文件）

因为是顺序型爬虫，且随着类目级别增多，爬的时间也变多，分了很多只，本应顺序爬取，但是每一级类目都是依靠上一级，所以可以同时开启

存储文件架构如下：

1. 首页抓下来，提取所有一级类目保存文件名 oneurl.md

2. 根据oneurl.md抓取二级类目，保存在2urls文件夹下，命名：

2urls

1-url.md 第一个一级类目下的二级类目们

2-url.md 第二个一级类目下的二级类目们

3.扫描2urls文件夹，读取所有文件，按文件中二级链接分别抓取三级类目，保持在3urls，命名:

3urls

1-1-url.md 表示1-url.md文件下的第一条链接的三级类目们

1-2-url.md 表示1-url.md文件下的第二条链接的三级类目们

以此类推

4urls

1-1-1-url.md 表示1-1-url.md文件下的第一条链接的四级类目们

1-1-2-url.md 表示1-1-url.md文件下的第二条链接的四级类目们

5urls

1-1-1-1-url.md 表示1-1-1-url.md文件下的第一条链接的五级类目们

1-1-1-2-url.md 表示1-1-1-url.md文件下的第二条链接的五级类目们

ausaspider.py抓取后存到oneurl.md,ausaspider1.py抓取后存到2urls

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

a = time.clock()

isdead = False

while not isdead:

try:

ausalogic("1-2")

isdead = True

except Exception as err:

logging.error(err, exc\_info=1)

b = time.clock()

logger.error('运行时间：' + timetochina(b - a))

ausaspider2.py，ausaspider3.py类目过多，循环检测上一级新增的类目

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

a = time.clock()

isdead = False

while not isdead:

try:

ausalogic("3-4")

except Exception as err:

logging.error(err, exc\_info=1)

pass

time.sleep(3600)

logger.error("一分钟后又跑一次")

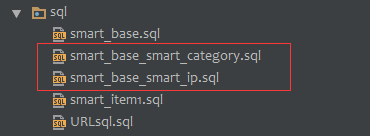
b = time.clock()

logger.error('运行时间：' + timetochina(b - a))

最后一级太多了，使用了并行fastspider.py，可根据源代码将其他级别也设置为并行

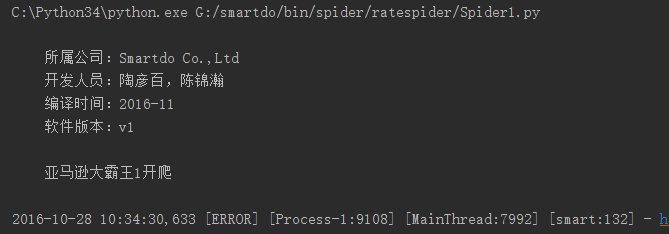
if \_\_name\_\_ == "\_\_main\_\_":

fastlevel5(num=25)

直接导入数据库文件，不必重跑

### 3.2 ratespider排名爬虫

运行截图如下：需先跑redismaster.py填充IP池



程序解释如下：

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

print(copyright("亚马逊大霸王1开爬"))

a = time.clock()

# 大类名

try:

# changeconfig("catchbywhich","bigpname") # 按类目来select类目

# category = getconfig()["catchurl"]

changeconfig("catchbywhich", "database") # 按数据库来select类目

changeconfig("redispoolname","ippool1")

changeconfig("redispoolfuckname","ippoolfuck1")

# category = ["15", "16", "17", "18", "19", "20", "21","22"]

category = ["9","10","11","12"] # 抓取这四个数据库所在的类目

except:

category = ["Appliances", "Arts\_ Crafts & Sewing"]

# 并行数量

try:

processnum = getconfig()["processnum"]

except:

processnum = 10

# 类目抓取数量

try:

limit = getconfig()["urlnum"]

except:

limit = 60

# 开抓，ko

ratelogic(category, processnum, limit)

b = time.clock()

logger.error('运行时间：' + timetochina(b - a))

## 4. 辅助工具使用bin/tool

### 4.1 proxyfiletool.py

处理代理IP地理位置

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

savetofile(filepath="config/base/IPtemp.txt", savepath="config/base/IP.txt")

先往config/base/IPtemp.txt按行填入代理IP，如

146.148.157.225:808

146.148.157.224:808

执行后会在config/base/IP.txt，发现

146.148.157.225:808-美国加利福尼亚州洛杉矶

146.148.157.224:808-美国加利福尼亚州洛杉矶

可手动在IP.txt填入。

### 4.2 proxymysqltool.py

将代理IP存入数据库(目前IP抽取都在数据库，也可从文件抽取）

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

allconfig = getconfig()

try:

config = allconfig["basedb"]

# config = {"host": "localhost", "user": "root", "pwd": "6833066", "db": "smart\_base"}

savetomysql(filepath="config/base/IP.txt",config=config)

except:

raise Exception("数据库配置未填")

将config/base/IP.txt中的代理IP存到数据库

### 4.3 createtabletool.py

数据库建表

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

for i in range(1,23):

db = str(i)

allconfig = getconfig()

try:

baseconfig = allconfig["basedb"]

tables = selecttable(baseconfig, db)

print(tables)

db = "db" + db

dbconfig = allconfig[db]

database = allconfig[db]["db"]

createtable(dbconfig, database, tables)

except:

raise Exception("数据库配置出错")

select类目数据库中字段database为1-23的记录，查找配置文件拼接db，形成db1,db2，得到真实数据库地址后开始建表：

"dbprefix":"db",# 类目数据库字段的前缀，如1，拼接后db1，找到下面真实的数据库地址

"db1": {"host": "45.41.88.189", "user": "root", "pwd": "smart2016", "db": "smartdb1"},

"db2": {"host": "45.41.88.189", "user": "root", "pwd": "smart2016", "db": "smartdb2"},

按类目数据库字段id命名数据库表，如1-1-1-1

### 4.4 urlspidertool.py

将抓取的类目URL汇总保存在config/base/URL.txt并存入类目数据库（直接导数据库文件，可不运行）

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

# 处理URL

dealurlfile()

# 保存入数据库

# config = {"host": "localhost", "user": "root", "pwd": "6833066", "db": "smart\_base"}

allconfig = getconfig()

try:

config = allconfig["basedb"]

keeptomysql(config=config)

except:

raise Exception("数据库配置出错")

URL.txt内容

1-10-2,Washers,https://www.amazon.com/Best-Sellers-Appliances-Clothes-Washing-Machines/zgbs/appliances/13397491/ref=zg\_bs\_nav\_la\_2\_2383576011,1-10,3,1,Appliances,1

1-10-3,All-in-One Combination Washers & Dryers,https://www.amazon.com/Best-Sellers-Appliances-Combination-Washers-Dryers/zgbs/appliances/13755271/ref=zg\_bs\_nav\_la\_2\_2383576011,1-10,3,1,Appliances,1

1-10-4,Stacked Washer & Dryer Units,https://www.amazon.com/Best-Sellers-Appliances-Stacked-Washer-Dryer-Units/zgbs/appliances/2399957011/ref=zg\_bs\_nav\_la\_2\_2383576011,1-10,3,1,Appliances,1

按上面列表插数据库

sql = 'insert into smart\_category (`id`,`url`,`name`,`level`,`pid`,`createtime`,`bigpname`,`bigpid`,`ismall`) values("{id}","{url}","{name}",{level},"{pid}",CURRENT\_TIMESTAMP,"{bigpname}","{bigpid}",{ismall}) on duplicate key update updatetime = CURRENT\_TIMESTAMP'

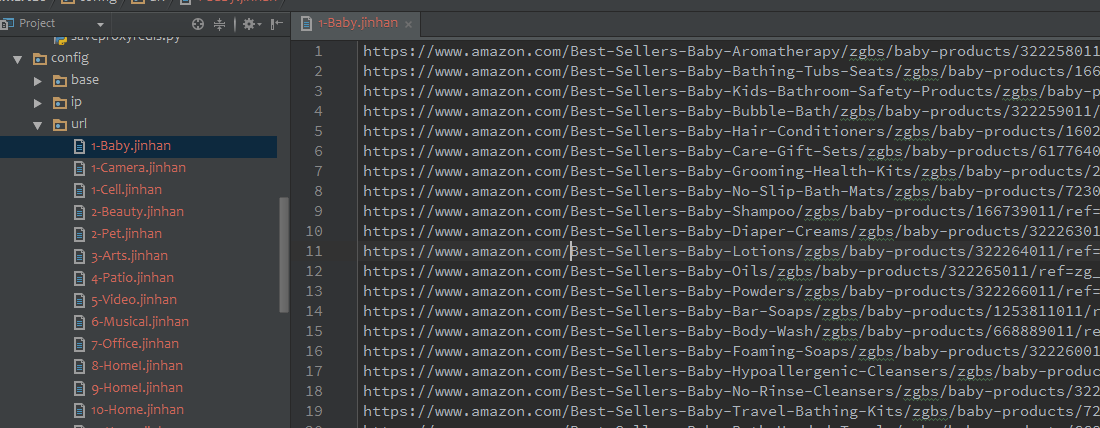
insertsql = sql.format(id=sqlzone[0], url=sqlzone[2], name=sqlzone[1], level=sqlzone[4], pid=sqlzone[3],

bigpname=sqlzone[6], bigpid=sqlzone[5], ismall=sqlzone[7])

### 4.5 validurlspidertool.py

可使用spliturltool.py切割文件

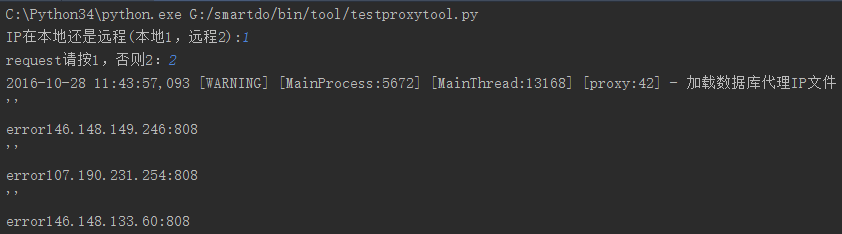
将config/url/\*.jinhan的类目链接,按行提取更改数据库字段database并设置标记位isvalid为1，示意该类目抓取并存到哪里.



上面类目将抓取并存在db1,db2数据库中，需要查config.json找到真正数据库。1-Baby.jinhan表示该文件类目链接抓取后存在数据库1，总共有22个数据库。

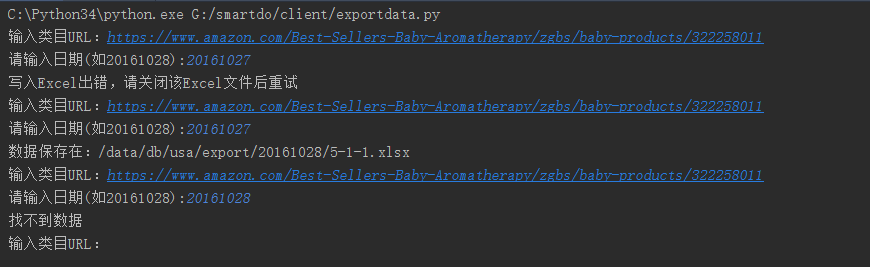
### 4.6 testproxytool.py

代理测试



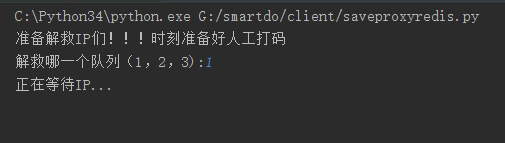
## 5. 客户端工具

### 5.1 client/exportdata.py 导数据



输入类目和时间即可导出Excel数据，所有数据保存在配置文件config.json datadir指定的位置

### 5.2 client/saveproxyredis.py 拯救IP



## 6. 外部文件数据存储架构

datadir = "G:"

将保存在C盘，且

detail （商品详情数据）

---2015 (年份)

---2016

---Arts\_ Crafts & Sewing (大类名）

---20161017 （日期）

---20161018

---3-1-1-1 （小类ID，小类名太长）

---3-1-1-2

---1-B00J5QM832.html （具体网页，可选择不保存，存在该文件不存在md文件，解析该文件）

---1-B00J5QM832.md （存入数据库之后的json字符串，见后文,如果存在该文件跳过）

---2-B03J5QM832.html （命名：小类排名-Asin)

items （商品列表数据）

---2015 (年份)

---2016

---Arts\_ Crafts & Sewing (大类名）

---20161017 （日期）

---20161018

---3-1-1-1 （小类ID，小类名太长）

---3-1-1-2

---1.md(第几页列表页）

---2.md（数据见下文）

---5.md

rateurl (类目链接数据）

export （导出的EXCEL文件）

---20161018

--- 20161019-170256.xlsx

详情页json 1-B00J5QM832.md

{

"asin": "B007QNFP40",

"bigname": "Arts\_ Crafts & Sewing",

"commentnum": 166,

"commenttime": "",

"id": "20161019-1-B007QNFP40",

"name": "Beading Kits",

"price": 12.59,

"rank": 835,

"score": 4.4,

"shipby": "FBA",

"smallrank": 1,

"soldby": "Amazon.com",

"tablename": "3-1-1-3",

"title": "Beadery Bead Extravaganza Bead Box Kit- 19.75-Ounce- Pearl",

"url": "https://www.amazon.com/dp/B007QNFP40"

}

列表页数据 1.md

41,B00NCV0C8E,New Design Antique Bronze Plated Blan...

42,B01BLVH54U,Disney Princess Melty Beads (1000 Beads)

43,B019MYI268,Ship From US Silicone Bracelets Exped...

44,B005E0CEI2,Bead Mats 2/Pkg-7.75"X7.75"

45,B000RB3EWS,2 Diamond Gemstone Sorting Tray Pearl...

46,B0026HT68M,Beadalon Bead Mats 3/Pkg 9-Inch by...

47,B000I6H4VE,Darice Large 3-Channel Flocked Bead B...

48,B01HF4AFTO,Neon Silicone bracelet Bangles Perfec...

# 四．服务器规划

VMware vSphere软件虚拟化管理两台主机

| 账号 | 密码 |
| --- | --- |
| vcenter.bestman.win（45.41.89.188） | bestman\smart002 Smart2016 |

| 服务器名称 | IP | 配置（硬盘/内存） | 用途 | 安装方式（Centos7) | 软件 |
| --- | --- | --- | --- | --- | --- |
| zhigan-01（主机2） | 45.41.88.187 | 300G+10G | 爬虫服务器 | 图形界面 | Mysql Python3 |
| zhigan-02（主机2） | 45.41.88.188 | 300G+8G | 爬虫服务器 | 图形界面 | Mysql Python3 |
| zhigan-03（主机2） | 45.41.88.189 | 300G+8G | Web服务器 | 图形界面  标注（图形界面消耗的内存大概600M，硬盘3.7G）|·  95G /home 200G /  4000M swap 250G /boot | Mysql Nginx Go 浏览器 |

# 五．服务器行为准则

所有服务器都是Centos,所以准则如下：

## 1. 基本命令

df -h 查看磁盘free -h 查看内存ip -s **addr** 查看ip地址ifconfig

## 2. 规范

(1)：文件结构

**mkdir** /data**cd** /data**app** -> /usr/**local**

ln -s /usr/**local** **app**

建立/data/app (放安装软件)

建立/data/www (放运行程序)

1. :日常操作

设置环境变量：

**cd** /etc/**profile**.**dvim** myenv.**shsource** myenv.**sh**

更改host

vim /etc/hosts

## 3. 维护

磁盘扩容 <http://www.centoscn.com/CentOS/config/2015/0315/4891.html>

centos7安装后，磁盘分了3个逻辑卷，

/dev/centos/root

/dev/centos/swap

/dev/centos/home

大部分磁盘空间都分给home了。

现在希望把空间分给root。

以下命令，通过system-storage-manager，删除home分区，把空间增加到root里。

（由于新装的系统，home下是空的，可以直接删除。

而且，由于home的文件系统是xfs，似乎只能扩容不支持缩减，所以只好删除。）

**# 安装ssm**

yum --disablerepo=\* --enablerepo=ustc\* install system-storage-manager

**# 查看分区**

ssm list

**# 卸载home**

umount /home

**# 删除逻辑卷home**

ssm remove /dev/centos/home

**# 查看释放出来的空间，并增加到root上**

ssm list

ssm resize -s +1.76T /dev/centos/root

**# 还需要使用xfs\_growfs扩容文件系统**

ssm list

xfs\_growfs /dev/centos/root

**# 最后，要把fstab中挂载home的一行删掉**

vi /etc/fstab

网路设置

ping 一个外网ip，比如114.114.114.114 看下通不，通的话就是dns的问题，ping不通那就是网关的问题

<http://www.cnblogs.com/visi_zhangyang/articles/2429185.html>

CentOS修改IP地址

**# ifconfig eth0 192.168.1.80**

这样就把IP地址修改为192.168.1.80(如果发现上不了网了，那么你可能需要把网关和DNS也改一下，后面会提到)，但是当你重新启动系统或网卡之后，还是会变回原来的地址，这种修改方式只适用于需要临时做IP修改。要想永久性修改，就要修改/etc/sysconfig/network-scripts/ifcfg-eth0这个文件，这个文件的主要内容如下（你的文件中没有的项，你可以手动添加）：

**# vi /etc/sysconfig/network-scripts/ifcfg-eth0**

DEVICE=eth0 **#描述网卡对应的设备别名**

BOOTPROTO=**static** **#设置网卡获得ip地址的方式，选项可以为为static，dhcp或bootp**

BROADCAST=192.168.1.255 **#对应的子网广播地址**

HWADDR=00:07:E9:05:E8:B4 **#对应的网卡物理地址**

IPADDR=12.168.1.80 **#只有网卡设置成static时，才需要此字段**

NETMASK=255.255.255.0 **#网卡对应的网络掩码**

NETWORK=192.168.1.0 **#网卡对应的网络地址，也就是所属的网段**

ONBOOT=yes **#系统启动时是否设置此网络接口，设置为yes时，系统启动时激活此设备**

CentOS修改网关

**# route add default gw 192.168.1.1 dev eth0**

这样就把网关修改为192.168.1.1了，这种修改只是临时的，当你重新启动系统或网卡之后，还是会变回原来的网关。要想永久性修改，就要修改/etc/sysconfig/network 这个文件，这个文件的主要内容如下（你的文件中没有的项，你可以手动添加）：

**# vi /etc/sysconfig/network**

NETWORKING=yes **#表示系统是否使用网络，一般设置为yes。如果设为no，则不能使用网络。**

HOSTNAME=centos **#设置本机的主机名，这里设置的主机名要和/etc/hosts中设置的主机名对应**

GATEWAY=192.168.1.1 **#设置本机连接的网关的IP地址。**

\*\*\*\*\*\*\*\*\*\*上面的文件修改完要重新启动一下网卡才会生效：**# service network restart \*\*\*\*\*\*\*\***

CentOS修改DNS

上面的都修改完之后，当你ping一个域名是肯能不通，但ping对应的IP地址是同的，这时我们需要修改一下DNS。修改DNS要通过修改/etc/resolv.conf这个文件：

**# vi /etc/resolv.conf**

nameserver 8.8.8.8 **#google域名服务器 nameserver 8.8.4.4 #google域名服务器**

通过上面的所有设置，系统应该可以上网了。

如果centos系统建立在虚拟机之上，那么在设置虚拟机的网络时请选择‘网桥适配器’连接。

# 六．英文介绍

# An Amazon Crawler

## Source Framework

Develope by Python, Look at the following：

spider (Crawler module)

--- download (Crawler Download Module)

--- parse (Crawler Parser Module)

--- logic (Crawler Logic Module)

bin (Crawler Execution File)

--- spider (Crawler Entrance)

--- tool (Auxiliary Tool)

config (Config Module)

--- base (Config File)

--- config.py

config.json

log.json

tool (Basic Tool)

--- jfile

--- jhttp

--- jjson

--- jmysql

--- log.py

acion (Action Module，Such as proxy IP,Useragent...)

test (Test Dir)

data (Data Keep)

log (Log Keep)

client (Export Data)

doc (Help Doc)

## Third Party Library (to be installed)

```

pip3 install xlsxwriter

pip3 install pymysql

pip3 install requests

pip3 install bs4

pip3 install redis

yum install libxslt-devel

pip3 install lxml

pip3 install -U selenium

pip install requests[socks]

```

## Setting Environment Variables

```

set PYTHONPATH="G:/smartdo" Window

export name="path" Linux

```