爬虫

- 自己爬的项目

  准备别人的源码吧

- 用的什么框架，为什么选择这个

  scrapy，只需要实现少量代码，就能够快速的抓取到数据内容。Scrapy 使用了 Twisted异步网络框架来处理网络通讯，可以加快下载速度，不用自己去实现异步框架，并且包含各种中间件接口，可以灵活的完成各种需求。 scrapy能够满足大量的需求，除非反爬虫特别变态，在考虑自己写框架来处理

- 框架问题：

- Scrapy的基本结构（5个部分，请求发出去的整个流程）

  引擎(Scrapy) - 用来处理整个系统的数据流处理, 触发事务(框架核心)

  调度器(Scheduler) - 用来接受引擎发过来的请求, 压入队列中, 并在引擎再次请求的时候返回. 可以想像成一个URL（抓取网页的网址或者说是链接）的优先队列, 由它来决定下一个要抓取的网址是什么, 同时去除重复的网址

  下载器(Downloader) - 用于下载网页内容, 并将网页内容返回给蜘蛛(Scrapy下载器是建立在twisted这个高效的异步模型上的)

  爬虫(Spiders) - 爬虫是主要干活的, 用于从特定的网页中提取自己需要的信息, 即所谓的实体(Item)。用户也可以从中提取出链接,让Scrapy继续抓取下一个页面

  项目管道(Pipeline) - 负责处理爬虫从网页中抽取的实体，主要的功能是持久化实体、验证实体的有效性、清除不需要的信息。当页面被爬虫解析后，将被发送到项目管道，并经过几个特定的次序处理数据。

- scrapy框架执行爬虫的流程？

引擎从调度器中取出一个链接(URL)用于接下来的抓取

引擎把URL封装成一个请求(Request)传给下载器

下载器把资源下载下来，并封装成应答包(Response)

爬虫解析Response

解析出实体（Item）,则交给实体管道进行进一步的处理

解析出的是链接（URL）,则把URL交给调度器等待抓取

- Scrapy的去重原理（指纹去重是个什么意思）
- Scrapy中间件有几种类，用过什么中间件
- scrapy中间件再哪里起的作用（面向切面编程）

代理问题：

- 为什么会用到代理
- 代理怎么使用（具体代码，请求在什么时候添加的代理）
- 代理失效了怎么处理

验证码处理：

- 登陆验证码处理
- 爬取速度过快出现的验证码处理
- 如何用机器识别验证码

模拟登陆问题：

- 模拟登陆流程
- cookie如何处理
- 如何处理网站传参加密的情况

分布式:

- 分布式原理
- 分布式如何判断爬虫已经停止了
- 分布式去重原理

数据存储和数据库问题

- 关系型数据库和非关系型数据库的区别
- 爬下来数据你会选择什么存储方式，为什么
- 各种数据库支持的数据类型，和特点，比如：redis如何实现持久化，mongodb是否支持事物等。。

Python基础问题

- python2和python3的区别，
- 如何实现python2代码迁移到python3环境
- python2和python3的编码方式有什么差别（工作中发现编码问题还是挺让人不爽的）
- 迭代器，生成器，装饰器
- python的数据类型
- Python常用的数据类型
- Python中单引号 双引号 三引号的区别
- 如何在一个 function 里面设置一个全局的变量？
- 如果 custname 字符串的内容为 utf-8 的字符，如何将 custname 的内容转为 gb18030 的字符串？
- 请写出一段 Python 代码实现删除一个 list 里面的重复元素。】
- 这两个参数是什么意思：args，*kwargs？
-

- 对__if__name__ == ‘main’的理解陈述

> __name__是当前模块名，当模块被直接运行时模块名为_main_，也就是当前的模块，当模块被导入时，模块名就不是__main__，即代码将不会执行。

- python是如何进行内存管理的？

> a、对象的引用计数机制 python内部使用引用计数，来保持追踪内存中的对象，Python内部记录了对象有多少个引用，即引用计数，当对象被创建时就创建了一个引用计数，当对象不再需要时，这个对象的引用计数为0时，它被垃圾回收。 b、垃圾回收 1>当一个对象的引用计数归零时，它将被垃圾收集机制处理掉。 2>当两个对象a和b相互引用时，del语句可以减少a和b的引用计数，并销毁用于引用底层对象 的名称。然而由于每个对象都包含一个对其他对象的应用，因此引用计数不会归零，对象也不会销毁。（从而导致内存泄露）。为解决这一问题，解释器会定期执行一个循环检测器，搜索不可访问对象的循环并删除它们。 c、内存池机制 Python提供了对内存的垃圾收集机制，但是它将不用的内存放到内存池而不是返回给操作系统。 1>Pymalloc机制。为了加速Python的执行效率，Python引入了一个内存池机制，用于管理 对小块内存的申请和释放。 2>Python中所有小于256个字节的对象都使用pymalloc实现的分配器，而大的对象则使用 系统的malloc。 3>对于Python对象，如整数，浮点数和List，都有其独立的私有内存池，对象间不共享他们的内存池。也就是说如果你分配又释放了大量的整数，用于缓存这些整数的内存就不能再分配给浮点数。

- Python里面如何拷贝一个对象？（赋值，浅拷贝，深拷贝的区别）

> 赋值（=），就是创建了对象的一个新的引用，修改其中任意一个变量都会影响到另一个。
>
> 浅拷贝：创建一个新的对象，但它包含的是对原始对象中包含项的引用（如果用引用的方式修改其中一个对象，另外一个也会修改改变）{1,完全切片方法;2，工厂函数，如list();3，copy模块的copy()函数}
>
> 深拷贝：创建一个新的对象，并且递归的复制它所包含的对象（修改其中一个，另外一个不会改变）{copy模块的deep.deepcopy()函数}

- 介绍一下except的用法和作用？

> try…except…except…else… 如果所有的except都不匹配，则异常会传递到下一个调用本代码的最高层try代码中。 try下的语句正常执行，则执行else块代码。如果发生异常，就不会执行如果存在finally语句，最后总是会执行。

- Python中__new__与__init方法的区别

> **new**:它是创建对象时调用，会返回当前对象的一个实例，可以用_new_来实现单例 **init**:它是创建对象后调用，对当前对象的一些实例初始化，无返回值

协议问题：

- http协议，请求由什么组成，每个字段分别有什么用,https和http有什么差距
- 证书问题
- TCP,UDP各种相关问题

- http、https协议有什么区别？

> http协议是超文本传输协议，被用于在web浏览器和网站服务器之间传递信息，以明文方式发送内容，不对数据加密，很容易被黑客入侵，安全性不高 为了数据传输的安全，https在http的基础上加入了SSL协议，SSL依靠ca证书来验证服务器的身份，为浏览器和服务器之间的通信加密

- http状态码？

> 表示网页服务器http响应状态的3位数字代码 2开头（请求成功）表示成功处理了请求的状态代码 3开头（请求被重定向）表示要完成请求，需要进一步操作 4开头（客户端错误）这些状态代码表示请求可能出错，妨碍了服务器的处理 5开头（服务器错误）这些状态代码表示服务器在尝试处理请求时发生内部错误

- 常见状态码：

> 200（成功）服务器已成功处理了请求 403（禁止）服务器拒绝请求 404（未找到）服务器找不到请求的网页 408（请求超时）服务器等候请求时发生超时

- 爬虫协议

> Robots协议（也称为爬虫协议、爬虫规则、机器人协议等）也就是robots.txt，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。Robots协议是网站国际互联网界通行的道德规范，其目的是保护网站数据和敏感信息、确保用户个人信息和隐私不被侵犯。因其不是命令，故需要搜索引擎自觉遵守。

数据提取问题：

- 主要使用什么样的结构化数据提取方式，可能会写一两个例子
- 正则的使用
- 动态加载的数据如何提取
- json数据如何提取

- 常用的网络数据爬取方法

> 正则表达式
>
> Beautiful Soup
>
> Lxml

- 什么是爬虫？

> 请求网站并提取数据的自动化程序

- 爬虫基本流程？

> 发起请求（scrapy发送get、post请求），可能包含请求头等信息，等待服务器相应 获取服务器响应内容，可能是网页文本（html、json代码），图片二进制、视频二进制等 解析内容（正则、xpath、json解析等 ） 保存数据（本地文件、数据库等）

- 遇到过什么反爬虫措施，如何解决？

- 对部分数据进行加密处理的(数据是乱码)

> 对部分数据进行加密的，可以使用selenium进行截图，使用python自带的pytesseract库进行识别，但是比较慢最直接的方法是找到加密的方法进行逆向推理。

- 基于用户行为，同一个ip段时间多次访问同一页面

> 利用代理ip，构建ip池 / 使用多个代理ip进行抓取或者设置抓取的频率降低一些，

- 请求头里的user-agent

构建user-agent池（操作系统、浏览器不同，模拟不同用户）

- 动态加载（抓到的数据和浏览器显示的不一样），js渲染

  模拟ajax请求，返回json形式的数据 selenium / webdriver 模拟浏览器加载（chromedriver安装） 动态网页的可以使用selenium + phantomjs 进行抓取

- 通过headers反爬虫

  对于基本网页的抓取可以自定义headers,添加headers的数据

- urllib 和 urllib2 的区别

  urllib 和urllib2都是接受URL请求的相关模块，但是urllib2可以接受一个Request类的实例来设置URL请求的headers，urllib仅可以接受URL。urllib不可以伪装你的User-Agent字符串。 urllib提供urlencode()方法用来GET查询字符串的产生，而urllib2没有。这是为何urllib常和urllib2一起使用的原因。

- 设计一个基于session登录验证的爬虫方案
- 列举网络爬虫所用到的网络数据包，解析包

  网络数据包 urllib、urllib2、requests 解析包 re、xpath、beautiful soup、lxml

- Python在服务器的部署流程，以及环境隔离
- Django 和 Flask 的相同点与不同点，如何进行选择？
- 写一个Python中的单例模式
- Linux部署服务脚本命令(包括启动和停止的shell脚本)
- 你用过多线程和异步嘛？除此之外你还用过什么方法来提高爬虫效率？

  scrapy-redis 分布式爬取 对于定向爬取可以用正则取代xpath

- POST与 GET的区别

  GET数据传输安全性低，POST传输数据安全性高，因为参数不会被保存在浏览器历史或web服务器日志中； 在做数据查询时，建议用GET方式；而在做数据添加、修改或删除时，建议用POST方式； GET在url中传递数据，数据信息放在请求头中；而POST请求信息放在请求体中进行传递数据； GET传输数据的数据量较小，只能在请求头中发送数据，而POST传输数据信息比较大，一般不受限制； 在执行效率来说，GET比POST好

- 什么是lambda函数？它有什么好处?

  lambda 表达式，通常是在需要一个函数，但是又不想费神去命名一个函数的场合下使用，也就是指匿名函数 lambda函数：首要用途是指点短小的回调函数

- 如何提高爬取效率？

> 爬虫下载慢主要原因是阻塞等待发往网站的请求和网站返回 采用异步与多线程，扩大电脑的cpu利用率； 采用消息队列模式 提高带宽

- request请求（封装http请求）方式中的post、get有什么区别？

> GET一般用于获取/查询资源信息，而POST一般用于更新资源信息 get是在url中传递数据，数据放在请求头中，post是在请求体中传递数据 get安全性非常低，post安全性较高，但是get执行效率却比Post方法好

- xpath、css选择器及返回类型区分？

> response.selector.xpath(css) 为了方便，其中的selector可以省略 返回：由selector组成的list，每个元素都是一个selector对象 1、SelectorList类型 case = response.xpath('//[@class (https://github.com/class)="content"]/ul/li') 2、*List类型 case = response.xpath('//*[@class (https://github.com/class)="content"]/ul/li').extract() 3、str类型 case = ''.join(response.xpath('//*[@class (https://github.com/class)="content"]/ul/li').extract()) extract()[0]选取第一个元素， extract_first()能达到一样的效果

- 模拟登陆原理？

> 因为http请求是无状态的，网站为了识别用户身份，需要通过cookie记录用户信息（用户、密码），这些信息都会在手动登陆时记录在post请求的form-data里，那么在爬虫时候只需要将这些信息添加到请求头里即可。

- 验证码

> 可以将验证码下载到本地人工识别填入

- 分布式原理？

> 多台机器多个 spider 对多个 url 同时进行处理

- 是否了解线程的同步和异步？

> 线程同步：多个线程同时访问同一资源，等待资源访问结束，浪费时间，效率低
> 线程异步：在访问资源时在空闲等待时同时访问其他资源，实现多线程机制

- 是否了解网络的同步和异步？

> 同步：提交请求->等待服务器处理->处理完毕返回 这个期间客户端浏览器不能干任何事 异步: 请求通过事件触发->服务器处理（这是浏览器仍然可以作其他事情）->处理完毕

- 使用redis搭建分布式系统时如何处理网络延迟和网络异常？

> 由于网络异常的存在，分布式系统中请求结果存在“三态”的概念，即三种状态：“成功”、“失败”、“超时（未知）” 当出现“超时”时可以通过发起读取数据的操作以验证 RPC 是否成功（例如银行系统的做法） 另一种简单的做法是，设计分布式协议时将执行步骤设计为可重试的，即具有所谓的“幂等性”

- 谷歌的无头浏览器？

> 无头浏览器即headless browser，是一种没有界面的浏览器。既然是浏览器那么浏览器该有的东西它都应该有，只是看不到界面而已。 Python中selenium模块中的PhantomJS即为无界面浏览器（无头浏览器）:是基于QtWebkit的无头浏览器，

- 数据如何去重，清洗，存入数据库？
- 如何查找到二叉树两个节点的最低公共祖节点？
- 数据库
- 关系型数据库和非关系型数据库的区别？

> 关系型：MySQL、Oracle、SQL Server、DB2等 优势：支持复杂查询。可以用SQL语句方便的在一个表以及多个表之间做非常复杂的数据查询 事务支持。使得对于安全性能很高的数据访问要求得以实现

- 非关系型数据库优势

> 性能高。NOSQL是基于键值对的，可以想象成表中的主键和值的对应关系，而且不需要经过SQL层的解析，所以性能非常高 可扩展性。同样也是因为基于键值对，数据之间没有耦合性，所以非常容易水平扩展

- 数据库索引
- 类型

> （1）普通索引：没有任何限制 （2）唯一索引：不允许建立索引的列有重复值，但可以有空值 （3）主索引：特殊的唯一索引，不允许有空值 （4）候选索引：唯一性，可以有多个候选索引

- 优缺点

> 优点：加快数据查找的效率 缺点：占用磁盘空间 增加了插入和删除的操作时间。一个表拥有的索引越多，插入和删除的速度越慢，如要求快速录入的系统不宜建过多索引

- 索引实现方式

> B+树 散列索引 位图索引

- SQL里面设置复合索引与单个普通索引的区别？

  复合索引只对和索引中排序相同或相反的order by 语句优化 如果存在一个多列索引，任何最左面的索引前缀能被优化器使用。所以联合索引的顺序不同，影响索引的选择，尽量将值少的放在前面。

- 数据库视图？

  视图是从一个或多个表（视图）导出的表，视图与表不同，视图是一个虚表，即视图所对应的数据不进行实际存储，数据库中只存储视图的定义，在对视图的数据进行操作时，系统根据视图的定义去操作与视图相关联的基本表

- 优点

  简化了操作，把经常使用的数据定义为视图 安全性，用户只能查询和修改能看到的数据 逻辑上的独立性，屏蔽了真实表的结构带来的影响

- 缺点

  性能差 修改限制

- 数据仓库

  数据仓库是一个面向主题的、集成的、稳定的、反映历史变化的、随着时间的流逝发生变化的数据集合。它主要支持管理人员的决策分析。 数据仓库收集了企业相关内部和外部各个业务系统数据源、归档文件等一系列历史数据，最后转化成企业需要的战略决策信息。

- 数据库事务？

  数据库事务是指作为单个逻辑工作单元执行的一系列操作，要么完全执行，要么完全不执行

- MySQL用户权限、库权限、表权限的控制?

  用户权限：连接数据库需要用户名、密码 库权限： `#给用户hehe赋予操作test库的所有权限 grant all on test.* to hehe@'localhost' identified by '123456';`
  表权限： `#给用户hehe操作test库goods表的insert，select，update的权限 grant insert,select,update on test.goods to hehe@'localhost' identified`