架构设计

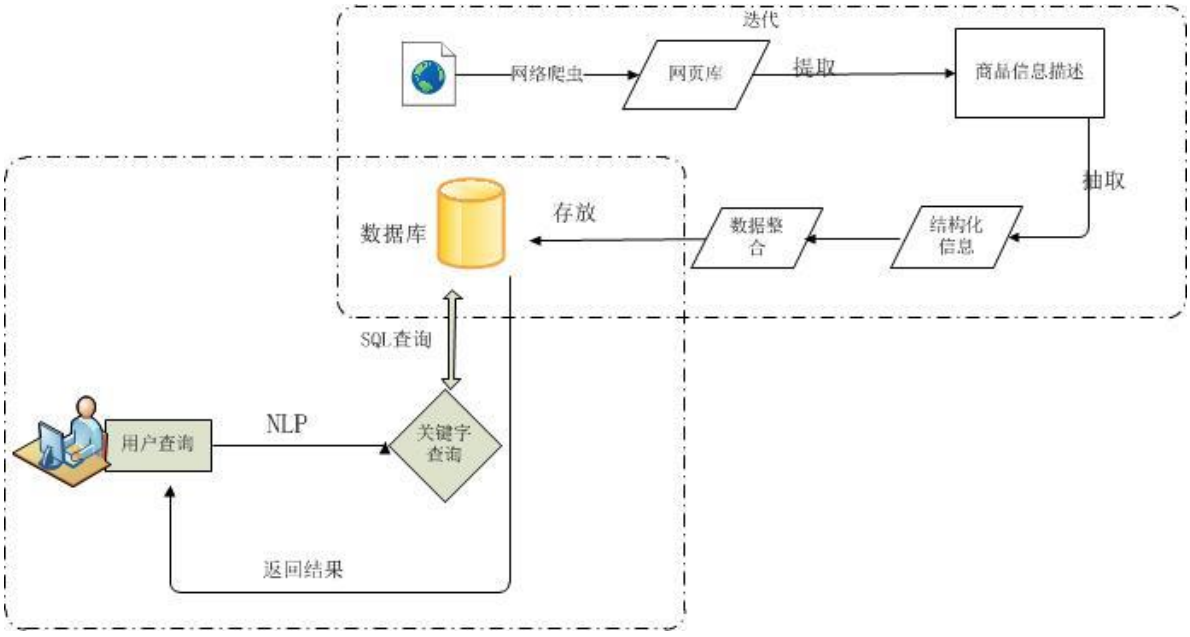
系统流程说明：

1、 网络爬虫从相应的电商网站的网页上抓取网页，通过解析 URL 对应的链接的商品网页从中获取商品信息并存放起来

2、 从获取的商品信息中通过去除无关的数据，抽取出结构化的数据并按照一定的格式存在的数据库对应的表格中

3、 用户输入查询关键字，按照这些关键字到数据库中进行匹配，找出相关的商品信息向用户展示出来。

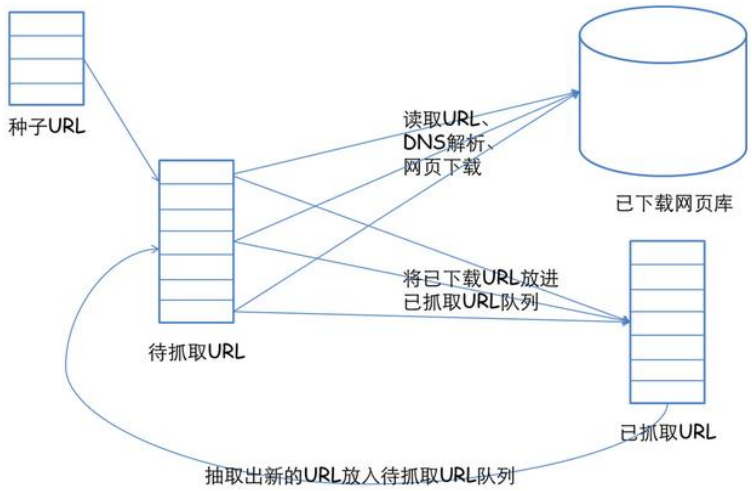
上面的每一个部分都对应着一个模块，现在我们对每个模块以及其功能做出具体分析。



第一模块：网络爬虫模块

网络爬虫的基本结构和架构如下所示：

一、网络框架如下：



基本工作的流程如下：

1） 首先确定一个种子 URL，然后找出要爬取内容的链接 URL；

2） 再将要爬取 URL 中的链接 URL 放入待爬取的 URL 队列中；

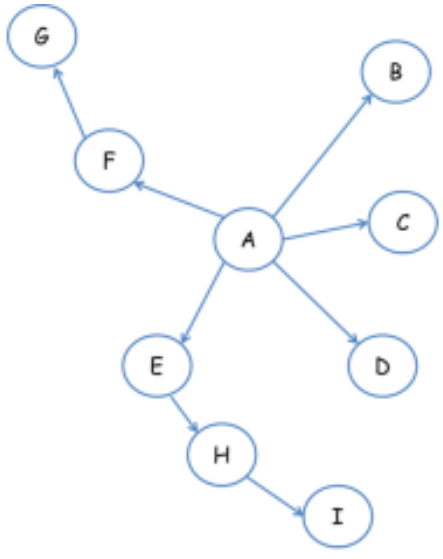
3） 从待爬取的 URL 队列中取出 URL，并将它们对应的网页爬取下来，利用页面解析，解析出我们需要的内容；

4） 分析以抓取的 URL 中的其他 URL，并将它们放入待抓取的 URL 中，从而进行下一次循环。

二、 抓取策略

在爬虫系统中，待抓取 URL 队列是很重要的一部分。待抓取 URL 队列中的 URL 以什么样的顺序排列也是一个很重要的问题，因为这涉及到先抓取那个页面，后抓取哪个页面。而决定这些 URL 排列顺序的方法，叫做抓取策略。

1、此次我们采用了深度优先遍历策略，深度优先遍历策略是指网络爬虫会从起始页开始，一个链接一个链接跟踪下去，处理完这条线路之后再转入下一个起始页，继续跟踪链接。我们以下面的图为例：



遍历路径：A-F-G　　E-H-I　　B　　C　　D

2、更新策略

网页的更新策略主要是决定何时更新之前已下载过的页面，常见的更新策略有以下三种：

1）　　历史参考页面

根据页面以往的历史更新数据，预测该页面未来何时会发生变化。一般来说，是通过泊松过程进行建模进行预测。

2）　　用户体验策略

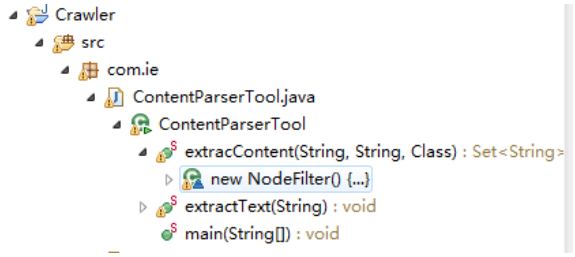
尽管搜索引擎针对于某个查询条件能够返回数量巨大的结果，但是用户往往只关注前几页结果。因此，抓取系统可以优先更新那些现实在查询结果前几页中的网页，而后再更新那些后面的网页。这种更新策略也是需要用到历史信息的。用户体验策略保留网页的多个历史版本，并且根据过去每次内容变化对搜索质量的影响，得出一个平均值，用这个值作为决定何时重新抓取的依据。

3）　　聚类抽样策略

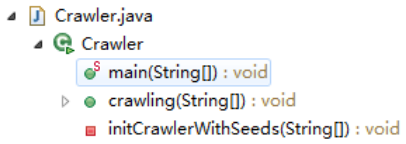
这种策略认为，网页具有很多属性，类似属性的网页，可以认为其更新频率也是类似的。要计算某一个类别网页的更新频率，只需要对这一类网页抽样，以他们的更新周期作为整个类别的更新周期。

三、　API 接口

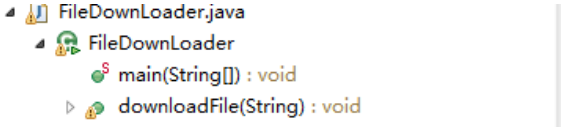
本次项目中接口的相关信息如下：



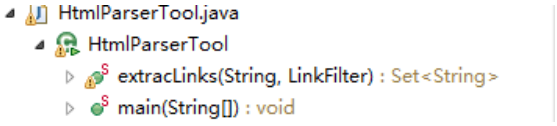
此类主要解析页面的内容，将爬取 的页面的内容中爬取出我们需要的内容，如书的作者，价格，简介等。



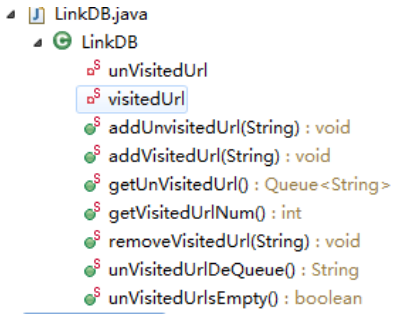
主类方法



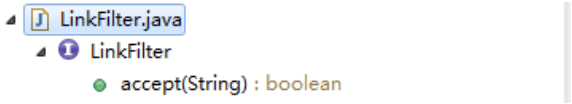
主要实现页面的内容的下载。



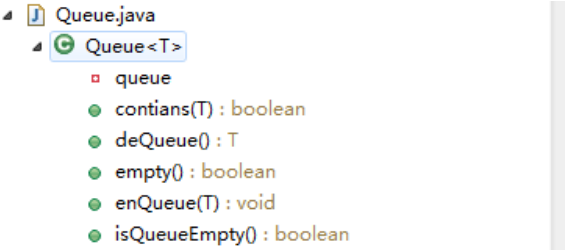
获取一个网站上的链接,filter 用来过滤链接



此类的主要功能是实现数据与数据库的连接，将数据传输到数据库中。



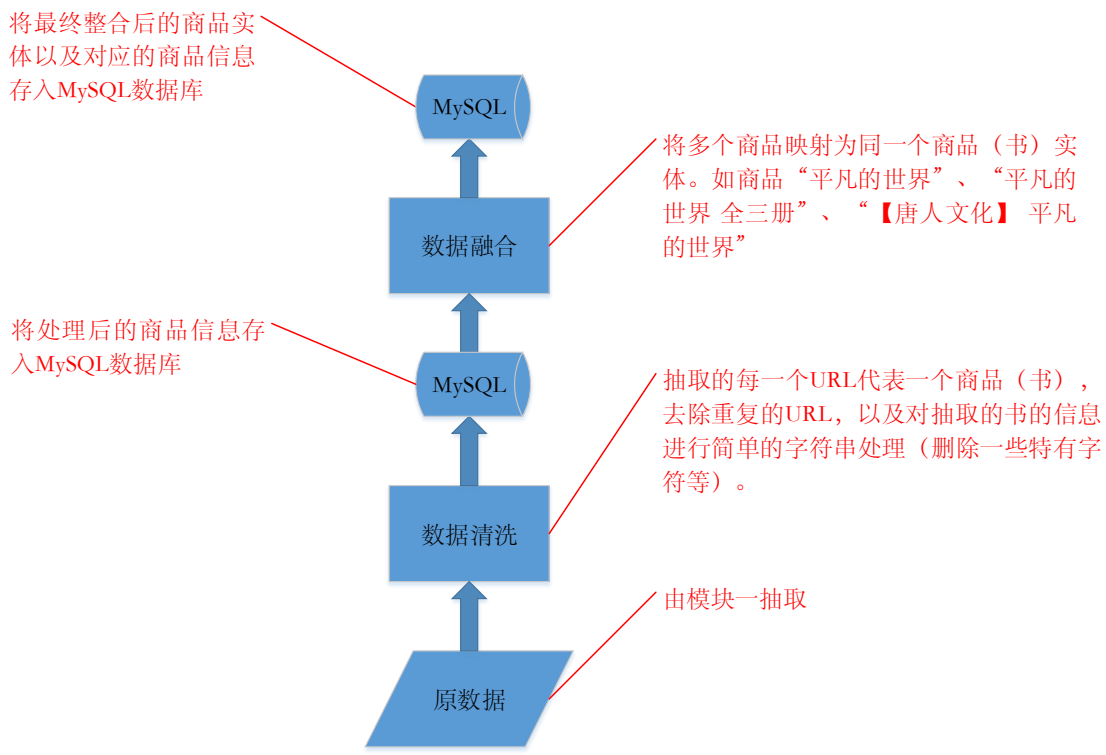
此功能主要实现网址 URL 的判断，判断此 URL 是否已经被爬取和待爬取的 URL，将这些 URL 分别放入不同的队列中。



此类的主要实现对列的基本方法。

第二模块：整合数据并存储到数据

模块二的数据流图：



在模块二中数据的存储需要进行两次，第一次存储是将抽取的数据清洗之后，第二次存储是最终融合后的商品数据。目前考虑两种存储都使用 MySQL。第一次存储的表数据结构为

表 1. 商品基本属性

属性	bookID	bookName	press	author	publishingTime	pagesCount	edition	…
数据类型	int	String	String	String	Date	int	String	…

表 2. 网页基本信息

属性	website	bookID	URLofBook	numOfBrowse	praise	badComment	URLID	price
数据类型	String	int	String	int	int	int	int	float

表设计理由：将商品信息和网站信息作为两个不同的实体，信息分别存放。便于存储，但对于查询，需要对数据进行整合后存于第二次的数据库表后进行查询操作。

第二次存储的数据表为

表 1. 商品实体与商品的对应表

属性	Index	bookName	bookIDs
数据类型	int	String	List

表 2. 商品的详细属性

属性	bookID	bookName	press	author	publishingTime	price	edition	…
数据类型	int	String	String	String	Date	float	String	…

表 3.商品实体的类别

属性	bookName	categories
数据类型	String	List

这样设计的理由是主要考虑到两种搜索类型，一种是直接使用书名，那么就根据书的名字查找表 1 商品实体表。找到和该书名对应的商品之后返回商品的详细信息。另一种是使用类别进行搜索，这就需要查找表 3 商品实体的类别表，通过类别的查找来锁定商品实体，然后再找到对应的商品。

在模块二中主要使用到的技术有相似度计算，instance match，ontology alignment 等。

第三模块：查询模块

在用户的查询中，我们使用自然语言进行处理，提取里面的关键字，这样我们就可以用基于关键字的方法从从数据库查询得到用户想要的信息,并按照一定的要求排序。