# 信息检索设计文档

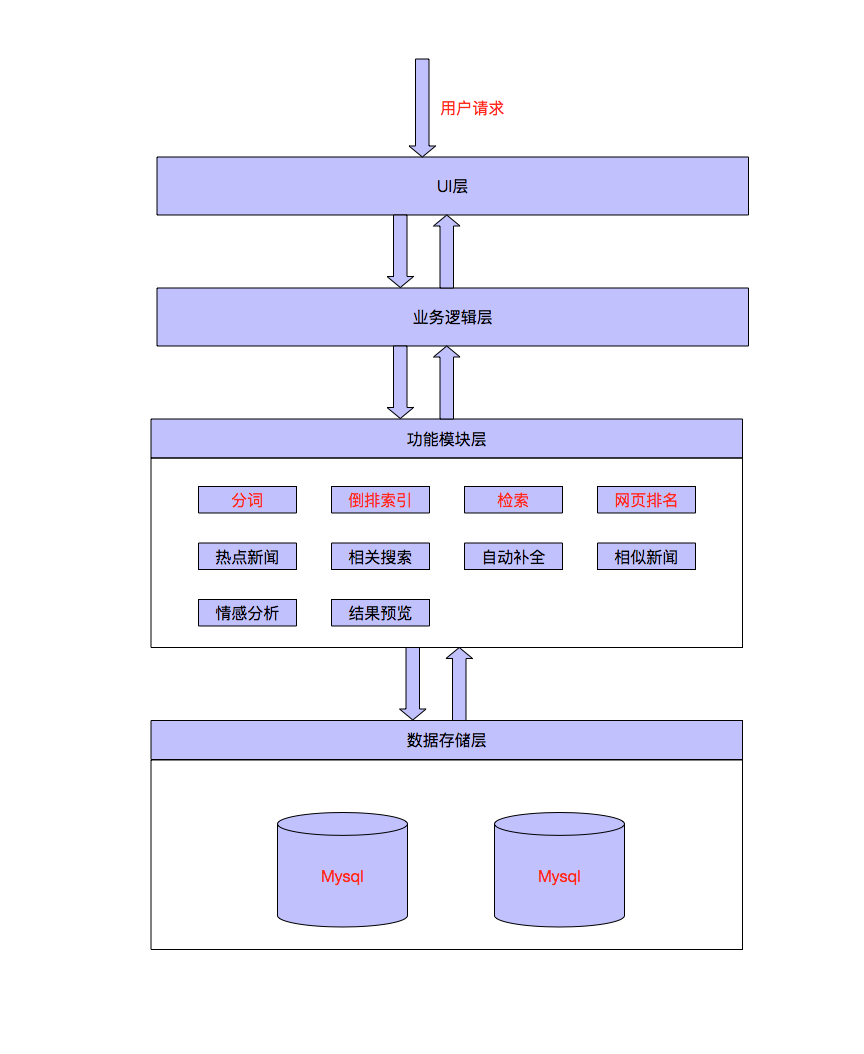
### 0.

数据库访问地址：mysql -P 124.16.81.231 -uroot -p123456

use information\_retrieval

github地址： https://github.com/LeoCui/NIR

### 1. 总体框架



### 2 存储方案设计

#### 2.1 mysql数据库

系统主要采取mysql来进行数据存储方案。mysql数据库中储存的信息包括文档内容信息，搜索历史记录信息，倒排记录表信息。

注：在操作数据库时，要注意查询的列有没有索引，没有索引的查询会非常慢，但是索引会增加空间。后期视代码逻辑决定要不要给某个列加索引。

建表语句见github上createTable.db文件

## 2.1.1 news\_info表

news\_info表存储的是新闻的基本信息，表中字段包括：

id： 自增id，主键

title: 新闻标题

url: 新闻的原始url

url\_hash: 对url进行hash

pv: 新闻浏览量

category: 新闻分类

is\_handled: 1表示已经被倒排记录表处理，0表示未被处理

comment\_number: 新闻的评论数

publish\_time: 新闻的发布时间

create\_time: 该条记录的创建时间

update\_time: 该条记录的更新时间

索引项：pv，url\_hash，comment\_number，publish\_time，这几项建索引是为了最热新闻推荐和rank。

## 2.1.2 content\_info表

content\_info表存储的是新闻的内容部分（不含评论），单独拆开一张表，是因为新闻内容的大小不确定，mysql中变量只能定长。为了节省内存，可能需要把一条新闻内容存在多条记录中。

表中字段包括：

id: 自增id，主键

news\_id: 所在的新闻的id

sequece\_number: 该条记录在该条新闻内容中的序列号。

content: 内容

create\_time: 该条记录中的创建时间

update\_time: 该条记录的更新时间

## 2.1.2 comment\_info表

comment\_info表存储的是新闻的评论，之所以要讲新闻和评论分开存储，原因如下：

a. 通常来说，一篇新闻的评论总是在更新（新增评论，删除评论）。但是新闻的内容是不变的，当评论更新时，我们就需要相应地更改倒排记录表，如果新闻内容和评论存储在一起，当评论更新时，我们需要对整个文档修改倒排记录表，分开的话只需要对评论部分修改倒排记录表。

b. 后续可能会有只针对评论信息的检索，那么对底层存储来说就需要将评论和新闻内容分开。

表中字段包括：

id: 自增id，主键

news\_id: 所在的新闻的id

comment\_number: 一条记录里的评论数

content: 评论内容，含有多条评论信息

create\_time: 该条记录中的创建时间

update\_time: 该条记录的更新时间

索引：news\_id

注：一条评论包括“日期 时间 用户名 内容”。评论之间用|分开

比如：2017-08-18 13:23:23 开飞机的贝塔 支持威武有希望了||2017-08-13 13：43:34杨总 哈哈

为了防止干扰，爬虫程序在向数据库中插入评论时，需要预处理，保证用户名和内容中不含空格和|

## 2.1.3 search\_history表

主要用于自动补全和相关搜索推荐

表中字段包括：

id：自增id，主键

content: 搜索内容

user\_id: 用户id，暂时不用

user\_ip: 用户ip，暂时不用

create\_time: 该条记录创建时间

update\_time: 该条记录更新时间

## 2.1.4 posting\_list表

倒排记录表，主要是为了将内存中建立的倒排记录表持久化存储起来，以便查询时使用。

表中字段如下：

id: 自增id，主键

term: 单词

term\_hash: 该单词对应的hash值，（crc32）

df: 出现该单词的文档的数目

sequence: 该条记录在该term对应的记录的序号

content: 倒排记录表项

create\_time: 该条记录创建的时间

update\_time: 该条记录更新的时间

由于term对应的文档数不定，无法确定posting\_list大小，所以可能一个单词对应的文档存储在多条记录中。

posting\_list中一个文档信息的结构如下： 0/1 docID tf <position1,position2>

不同文档之间用|分开其中:

0表示新闻（news\_info表），1表示评论（comment\_info表）

docID 表示文档ID

tf 表示term在docID中出现的次数

<>中保存的是位置信息，用于位置索引

如： 0 34 3 <1,5,8>|1 45 2 <3,4>

注：当content的内容不够一个容纳一个文档的位置信息时，会把该文档的位置信息拆成多份存储。后面用到的时候需要注意。

## 2.1.5 hot\_news表 热点新闻表，主要存储的是最近的最热新闻。

表中字段如下：

id：自增id，主键

news\_id：新闻在news\_info表中的id

rank：该条新闻的热度排名

create\_time：该条记录创建的时间

update\_time：该条记录更新的时间

## 2.1.6 dictionary表

词典表，存储的是所有的term

表中字段如下：

id: 自增id，主键

term: 词项

term\_hash: 词项的hash（crc32）

create\_time：该条记录创建的时间

update\_time：该条记录更新的时间

## 2.1.7 kgram\_index表

存储的是kgram索引，用来进行通配符查询

表中字段如下：

id: 自增id，主键

kgram: kgram索引

kgram\_hash：kgram的hash值（crc32）

number: 该条记录中content中termId的个数

content: content里面存的是该kgram对应的在dictionary表中的term的id的集合。格式为<termId1, termId2>，一条记录插不下需要拆成多条插入。

create\_time：该条记录创建的时间

update\_time：该条记录更新的时间

### 3. 模块及接口设计

#### 3.0 web服务模块

web后端：python + django

web前端：bootstrap + jsp

#### 3.1 爬虫模块

爬虫模块负责爬取网站新闻存入数据库中，可以爬取今日头条APP，网易新闻APP，新浪，搜狐等网站的数据。>10w个网页

移动端爬取方法：fidder抓包拿到移动端目的ip，然后模拟请求。

技术：python + requests + BeautifulSoup

注：为了防止多次爬取到相同的新闻，在news\_info表中设置了url\_hash这个字段，保证相同url的新闻只被加入数据库中一次。

## 3.1.1 初始化

这部分负责初始化爬取部分新闻，然后按照指定的格式存入数据库中的news\_info，content\_info，comment\_info表中，供建立倒排记录表使用。

## 3.1.2 增量

这部分主要是增量爬取新的新闻添加到数据库（news\_info，content\_info，comment\_info表）中。等项目稳定之后及时更新

主要包括一个天级别执行的定时脚本。

## 3.1.3 网页更新

这部分指的是网页内容有变化（评论增加，评论删除）的时候，用来更新数据库（news\_info，comment\_info表）。

这部分和增量模块不同，增量模块指的是爬取了新的网页，本模块指的是已经爬取的网页的内容改变。

当网页内容改变时（比如该网页被删除），除了更改news\_info，comment\_info表之外，还需要删除倒排记录表中含有原有docID的记录。这部分比较麻烦，前期可先不做。

#### 3.2 建立倒排索引模块

这部分指的是读取数据库中的网页，然后建立倒排记录表存入数据库中的posting\_list表中。

技术： java

## 3.2.1 分词

略

## 3.2.2 初始化建立倒排记录表

读取news\_info中的is\_handled为0的新闻，处理后建立倒排记录表，然后存入posting\_list表中。

因为该部分需要对外提供1个接口：

buildPostingList();

## 3.2.3 更新倒排记录表

更新倒排记录表主要包括2个部分：新增网页，删除网页。

更新倒排记录表有两种方法：1. 直接在原有的倒排记录表的基础上更新，2.建立另一个倒排记录表，我们暂时选用第一种方法。

该部分需要对外提供2个接口：

updatePostingList(): 读取newsInfo中的is\_handled=0的新闻，然后更新posting\_list表。

deleteDocInPostingList(docID)：当某个网页被删除时，需要将posting\_list表中含有该网页的记录删除，可先不做。

在外部有个定时脚本定时调用该接口去更新倒排记录表。

#### 3.3 检索模块

该部分主要接受用户的query，然后返回对应的docID的集合。检索使用hash表的方式。在倒排记录中有字段hash，表示某个term的hash值。

技术：java

该部分需要对外提供一个接口：

query 函数：

输入：json格式，设为input

input['query']: 查询的单词

input['page']；第几页

input['category']: 分类

input['source'] : 来源

input['from']: 开始时间，如2017-12-09 12:00:00

input['to'] : 结束时间

input['sort'] : 排序方式： 0表示按照相关度排序，1表示按照时间排序

输出： json格式，设为output

output['resultCount'] : 所有相关的文档的数量

output[' keywords']： 查询的关键词，是一个list

output['docList']: 相关的文档的信息，是一个list，list成员为doc

doc的结构如下：

doc['id'] : 文档的id

doc['kind']: 文档的类别，0表示内容，1表示评论

doc['relationship']：文档的相关度，如95.32%

#### 3.4 网页rank模块

这部分主要负责对检索到的文档打分，排序并输出。

技术：java

该部分需要对外提供一个接口：

rank(str，docIDSet，maxNumber): str表示查询的字符串，docIDSet表示相关的文档集合。maxNumber表示返回的文档数目，-1表示不限，返回值是排好序的文档集合。

#### 3.5 其他模块

## 3.5.1 首页热点新闻推荐

这部分主要包含一个定时脚本，定时更新hot\_news中的内容。

更新算法未定。

## 3.5.2 相关搜索推荐

该部分主要根据search\_history中的记录与当前query比较，找出相关的搜索结果。

## 3.5.2 搜索自动补全功能

这部分主要根据search\_history中的记录与当前输入比较，给出相关的提示。主要是前端的一段javascript代码来完成。

## 3.5.3 评论情感分析

## 3.5.4 结果预览

## 3.5.5 snippet生成

## 3.5.6 相似新闻推荐

## 3.5.7 更换背景