修订日志：

|  |  |  |
| --- | --- | --- |
| 修改日期 | 修改人 | 备注 |
| 2019/11/17 | 刘晨旭 | 添加部分内容 |
| 11.17 | 王文祥 | 数据格式化处理 |
| 11.18 | 李玫 | 增加elasticsearch部分 |
|  |  |  |
|  |  |  |

**数据格式化处理**

数据被爬取到本地后，应对数据进行格式化处理和保存，以便于其他模块保存和使用。

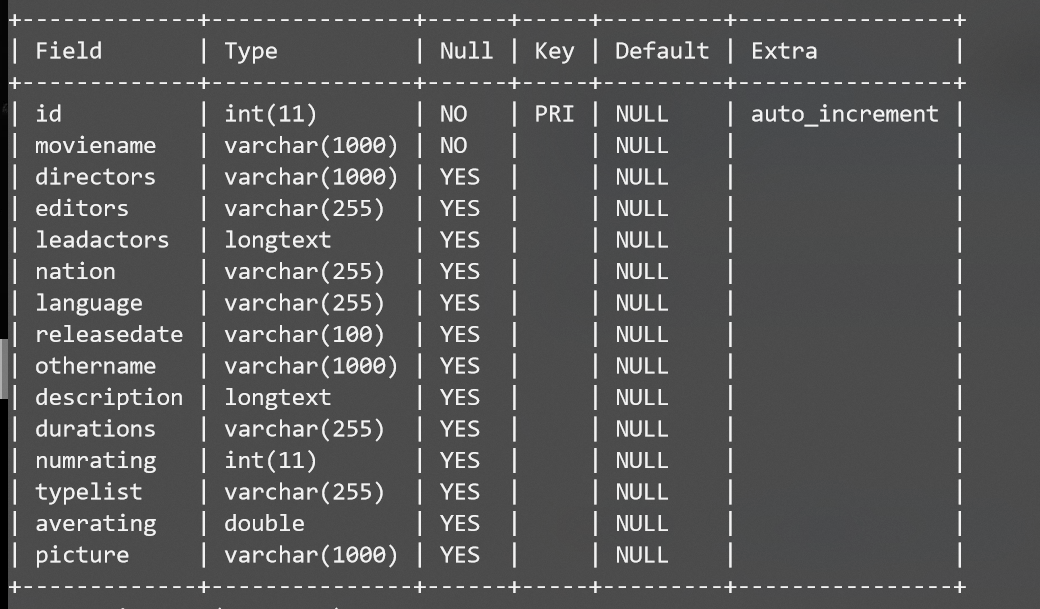
在开发过程中，每一条电影信息数据在流通的过程中都被按照片名、导演、编剧、主演、类型、制片国家/地区、语言、上映日期、片长、又名、简介、豆瓣id、平均评论得分、评论总人数、封面url保存，每一条评论数据在流通的过程中都被按照点赞数、用户链接、用户名、是否看过、推荐等级、打分、评论日期、评论内容、豆瓣id、电影名称的属性保存，并被相继保存为csv、json格式，以便于其他模块读取。

在数据的保存和入库过程中，我们遇到了一些问题。例如，有些外国演员的人名中包含有单引号（如O'Hara），有些电影简介和评论中包含有双引号，这就使得我们在入库语句的内容中不能直接使用单引号或双引号进行入库。针对这种情况，我们在入库语句中加入转义字符，这样入库语句在执行时就不会将其识别为含有特殊意义的元字符，而是认为它是一般字符，便可以顺利完成入库。

**原始数据到MySQL**

前一阶段爬虫爬到的数据是使用csv或json存储的文件，我们需要将其存储到MySQL数据库中。

以movie信息为例，由于数据爬取是在Windows操作系统中完成的，得到了一个csv文件，编码为‘gbk’编码，很不利与后续处理，需要将其转化为utf-8编码格式。尝试了多种方式之后，我们最后选择了一种简单的方式，通过Windows记事本的另存为utf-8的功能完成这个工作。观察该csv文件，我们建立了如下图所示的数据表



然后使用Python代码逐一读取csv文件，将每一行翻译为一条SQL插入语句，存入insert\_movie.sql文件中。在MySQL shell中，执行source insert\_movie.sql，将数据全部插入。

**MySQL到Hbase**

由于我们的系统部分功能需要使用Hbase数据库，需要将数据从MySQl导入到Hbase数据库中。我们使用Sqoop工具完成这一任务。

Sqoop是一款开源的工具，主要用于在Hadoop(Hive)与传统的数据库(mysql、postgresql...)间进行数据的传递，可以将一个关系型数据库（例如 ： MySQL ,Oracle ,Postgres等）中的数据导进到Hadoop的HDFS中，也可以将HDFS的数据导进到关系型数据库中。

以movie表为例，在hbase shell中先创建一个movie表，然后运行命令。

sqoop import --connect jdbc:mysql://47.92.101.31:3306/liuchenxu\_test?zeroDateTimeBehavior=CONVERT\_TO\_NULL --username 'liuchenxu' --password 'Cloud302@' --table 'movie' --hbase-table 'movie' --hbase-row-key 'id' --column-family 'region'

这样就将MySQL中movie数据表导入到了Hbase中，Hbase中movie的行键为movie的id。其他数据表类似。

# Elasticsearch

## 1数据库设计

### 1.1基本设置

Elasticsearch有自己独特的数据库结构，在ES中一个索引（index）是一个文档的集合，每个索引有唯一的名字，一个集群中有多个索引。在一个索引中可以索引不同类型的文档，称为“Type”，在高版本中已经不建议使用了。一个索引一个类型。每一条被索引的数据被称为document，是基本信息单元，用json的格式来表示。

首先创建索引，本次项目中设定的分片数shard为3，Replication为1。每一个分片本身也是能完善且独立的“索引”，并可以放置在任意的节点上。而副本可以提升访问压力过大时的处理请求能力，并且当分片所在的机器宕机，可以通过副本恢复避免数据丢失。

### 1.2字段设计

Es对于String类型的数据有两种存储方式，text和keyword。

**keyword：**keyword类型适用于索引结构化的字段，比如email地址、主机名、状态码和标签。如果字段需要进行过滤(比如查找已发布博客中status属性为published的文章)、排序、聚合。keyword类型的字段只能通过精确值搜索到。

**Text：**当一个字段是要被全文搜索的，比如Email内容、产品描述，应该使用text类型。设置text类型以后，字段内容会被分析，在生成倒排索引以前，字符串会被分析器分成一个一个词项。text类型的字段不用于排序，很少用于聚合。

根据数据的特点，为了节省空间并且增加查询的速度。设计了如下的表结构：

|  |  |  |
| --- | --- | --- |
| field含义 | field名称 | field类型 |
| 片名 | Name | Text |
| 导演 | Director | keyword |
| 编剧 | Writer | Keyword |
| 主演 | Actor | Keyword |
| 类型 | Type | Keyword |
| 制片国家/地区 | Country | Keyword |
| 又名 | alias | Text |
| 简介 | Brief | text |
| Id | Id | 直接作为doc的\_id |
| 平均得分 | score | float |
| 评论总人数 | Cnum | int |

在我们的数据集中，存在一个字段会包含多个值的情况，如一部电影有多个演员，或者有多个别名等。ES能够很便利地适配这一点，任何一个字段都可以包含0个或多个值。但是每个字段内的内容必须是同一个数据类型。

### 1.3 Analyzer设置：

Es中使用Analyzer把String转换为一串token和term。在索引时和查询时都需要用到。我们引入了IK分词器来对中文进行支持（ES自带的四种analyzer不支持对中文的分词处理）。IK分词有两种模式：ik\_max\_word和ik\_smart。

1、ik\_max\_word会将文本做最细粒度的拆分，比如会将“中华人民共和国人民大会堂”拆分为“中华人民共和国、中华人民、中华、华人、人民共和国、人民、共和国、大会堂、大会、会堂等词语。

2、ik\_smart会做最粗粒度的拆分，比如会将“中华人民共和国人民大会堂”拆分为中华人民共和国、人民大会堂。

为了索引时最大化地将文章内容分词，在搜索时更加精确地搜索到想要的内容，将在索引时使用ik\_max\_word，在搜索时采用ik\_smart。

## 2数据插入

Elasticsearch支持使用restful请求对数据库进行增删改查等操作。为了一次性将多条数据插入，使用ES的bulk方法，能够一次性操作多条增删改查语句。

因此，需要对数据进行预处理以满足ES特定的格式（bulk api对json的语法，有严格的要求，每个json串不能换行，只能放一行，同时一个json串和一个json串之间，必须有一个换行）。

通过python实现了数据的预处理，使用json库来实现字符串的生成，将原来为csv格式的数据处理以下格式(在这里省略部分字段展示)：

{"index": {"\_id": "26698000"}}

{"name": "心理罪", "director": "谢东燊", "writer": ["顾小白", "李昭", "曹晨", "雷米"], "type": ["动作", "悬疑", "犯罪"], "country": "中国大陆", "alias": ["心理罪大电影", "Guilty of Mind"], "brief": "一杯人血牛奶……", "score": "4.9", "cnum": "110712"}

可以看到，每一条插入操作需要一条index在前表明插入的doc的id，此id对应的就是数据库中的主键id，也是数据源中的id，便于数据库之间交通互联。紧接着的部分是不能包含换行符的，各个字段的内容。最后拼接上请求头：

POST movieinfo/\_bulk