**实验三 数据库实现设计**

**一、实验目的**

按照设计要求，把构件、类等组装成为最终可交付使用的信息系统工作。同时，功能的关键都是与数据库的连接，如何正确有效地连接到数据库，查找，添加，修改数据是此软件的关键，是各类数据库应用分析、开发、调试的全过程，并能以一个系统分析和设计员的身份参与项目，提高学生独立分析问题和解决问题的能力。

**二、实验内容**

1. 采用何种方式连接数据库，使数据库能够尽可能的适用于系统，方便开发。目标是正确有效地连接到数据库，能够对数据库进行有效地查询，添加，修改，删除操作。
2. 开展类的实现，编写类的程序代码，并放入到构件之中。包括生成类、类属性代码和类操作代码等。
3. 开展接口设计，其中包括外部接口、内部接口等。
4. 实施数据结构实现，包括逻辑结构实现、物理结构实现、数据结构与程序代码的关系。
5. 进行开展最重要的实现，数据库实现和界面实现。
6. 最后实现或者集成子系统，按照设计要求，把子系统组装成为最终可交付使用的信息系统工作。

**三、实验要求：**

包括确定系统的实现结构，子系统、类和接口的实现，单元测试，系统集成等。具体要求是需要确定实现结构，然后再制定实现的迭代计划。接下来通过多次迭代实现各个子系统和每一个子系统中的类和接口，并进行单元测试。并把每次迭代的结果交进行系统集成，通过多次迭代完成实现最终系统。

**四．设计流程和实现方案**

**1 引言**

**1.1 编写目的**

此份用户手册是面向于使用本电影数据分析系统的用户，本手册会详尽的说明本系统的功能及使用方法，用过阅读本手册用户会对本系统有一个整体全面的认识。

**1.2 背景**

A 软件系统的名称：电影数据分析系统

B 此软件项目的任务提出者：

开发者：王洋洋、何宗奎、姜斯文、李斌、杨俊

用户：电影爱好者

**1.3 术语**

|  |  |
| --- | --- |
| 缩写、术语 | 解 释 |
| MONGODB | 新型数据库、面向对象数据库 |
| DDB | 分布式数据库 |
| Nodejs | 一种运行在服务端的 JavaScript |

**1.4 参考资料**

1. 《计算机软件工程规范国家标准汇编》，中国标准出版社，2000.7.包含以下标准：
2. GB/T 11457-1995《软件工程术语》；
3. GB/T 8566-1995《信息技术软件生存期过程》；
4. GB/T 8567-1988 《计算机软件产品开发文件编制指南》；
5. GB/T 1526-1989 《信息处理产品数据流程图的文件编制符号及约定》；
6. GB/T 9385-1988 《计算机软件需求说明编制指南》；
7. 教育部教育信息化标准:JY/T 1001-2012《教育管理信息教育管理基础代码》。
8. Nodejs 入门教程。

**2 用途**

**2.1功能**

本软件是面向小型电影数据分析电影爱好者，及推荐电影所做成的。主要包括：数据爬取、数据维护、电影信息展示、电影推荐等模块。

数据爬取业务模块主要包括从猫眼电影数据进行爬取，并且将其存入数据库中。该业务获取的基础数据是整个系统的根基，关系到后续功能的实现。

数据维护主要是对其中的数据进行管理，能够保证数据能个得到及时且有效的管理。也就是进行相应的修改、添加、删除、查找等操作。

电影信息展示主要是按照用户的需求对所筛选出的电影信息进行对于的展示，方便用户进行阅读比如电影评分TOP10、电影票房TOP10、每月电影上映数量、每月电影票房、电影来源、最受欢迎电影TOP10、中外电影票房对比以及上映电影类型等信息。

电影评分TOP10主要是分析从猫眼电影爬取2018这一年上映的所有的电影进行对比，从中评出用户评分最高的10部电影；每月电影票房，就是将电影按照上映的时间进行分类，一共分为12个月，并将每个月上映的电影票房进行累加，然后横向进行比较电影。上映电影类型信息是将每一部电影类型进行统计，统计出2018年电影类型最多的那一种类型，也进一步反映市场对某一类型电影票房的反响。

**2.2性能**

**2.2.1 精度**

本软件基于图表进行展示，用户很容易就能看出各部电影的特点，几乎不用什么操作，我们对各个用户请求进行了封装，默认显示。所以用户打开系统就可以看到系统要展现给用户的结果。

**2.2.2 时间特性**

本软件使用的是面向对象的数据库，也是目前最流行的数据库mongoDB数据库，它是介于关系数据库和非关系数据库之间，查询功能非常强大。对用户的请求能够很快的进行响应，更新处理时间、数据传输、转换时间、计算时间都非常迅速。

**2.2.3 灵活性**

本软件各个模块的操作非常简单，灵便。在数据爬取、数据维护、数据展示各个模块中，相关的信息可以快速集中的输入以及进行查询，而各个模块间的转换和信息的增删改的操作，都可以通过系统管理员进行相关的操作。

**2.3 安全保密**

本软件在安全保密上设置的比较细致，因为该系统的数据库比较重要，以及数据库的登录进行修改、增加、删除等数据库操作的时候，需要系统管理员提供正确的密码及用户名才可以进行操作，因此可以防止数据库里的信息被丢失。保证了数据库的安全性。

**3 运行环境**

**3.1硬件平台**

运行本人家所要求的硬设备的最小配置：

1. Intel i5 处理器、内存1G；
2. I/O设备；显示器、鼠标、键盘；

**3.2支持软件**

说明为运行本软件所需要的支持软件，如

1. 操作系统：windows7及以上版本；
2. 支撑框架：服务器使用的是基于Nodejs 的Express服务器搭建框架，python3.0、java8.0 等环境；
3. 数据库：Mongodb4.0,Npm3.10 模块管理；
4. 开发工具：pycharm，vscode；

**3.3 数据库**

为支持本软件的运行所需要的数据库：MongoDB4.0

**4 软件设计**

**4.1 数据爬取模块**

4.1.1.爬取目标

首先确认要爬取的电影数据，该系统默认爬取网页数据是猫眼电影2018年上映的电影数据如（图1 猫眼电影索引）

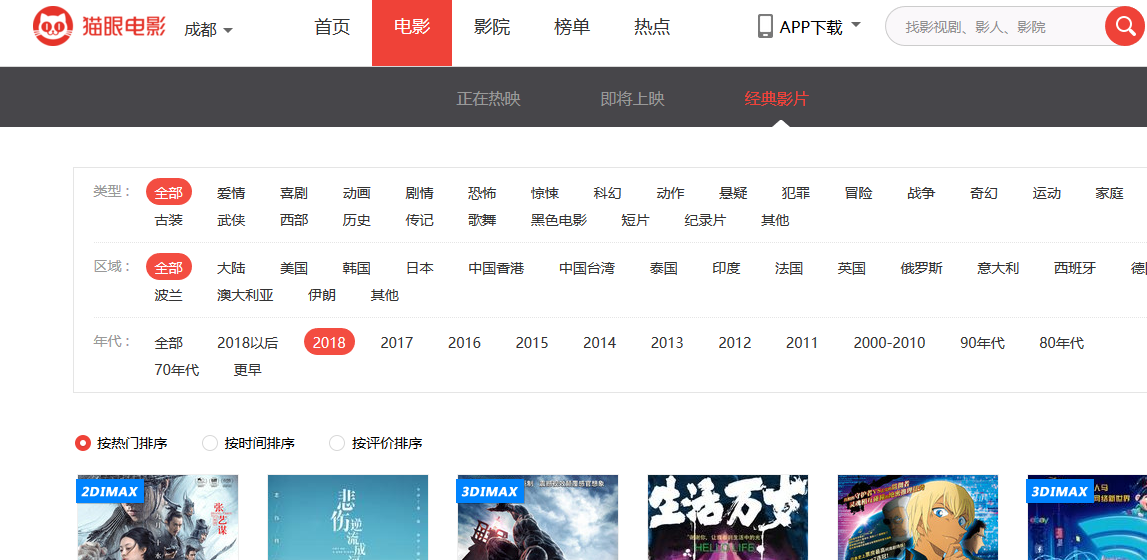


图 1 猫眼电影索引

我们的目标就是获取索引页里所有电影链接以及评分信息。

4.1.2 详情页

对详情页的信息进行获取，主要获取内容是电影名称、类型、国家、时长、上映时间、评分、评分人数，累计票房。



图 2 详情页信息

4.1.3 反爬虫破解

通过开发工具发现，猫眼针对评分，评分人数、累计票房的数据，施加了文字反爬。



图 3 文字反爬

通过分析文字反爬中的js事件，我们可以看出只要刷新页面，三处文字编码就会改变，无法直接匹配信息。所以需要下载文字文件，对其进行双匹配。然后将woff格式转换为xml 格式，以便在pycharm 中查看详细信息。

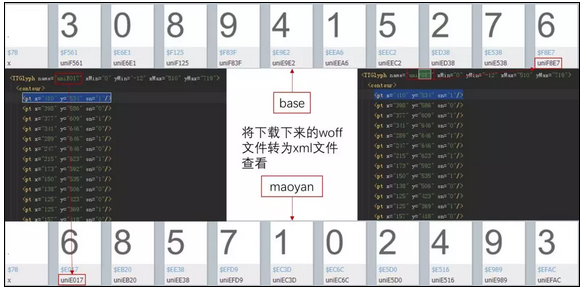


图 4 文字反爬破解

文字反爬破解后，接下来的工作就简单了；获取自己需要的数据，构造请求头、获取电影详情页链接、获取电影详情页的信息等等。

4.1.4 数据存储

我们要把获取的数据保存到数据库中。我们使用的是MongoDB数据库，它是转为可扩展性、高性能、高可用性而设计的数据库，它可以从单服务器部署扩展到大型、复杂的多数据中心架构。利用内存计算的优势，MongoDB能够提供高性能的数据读写操作。

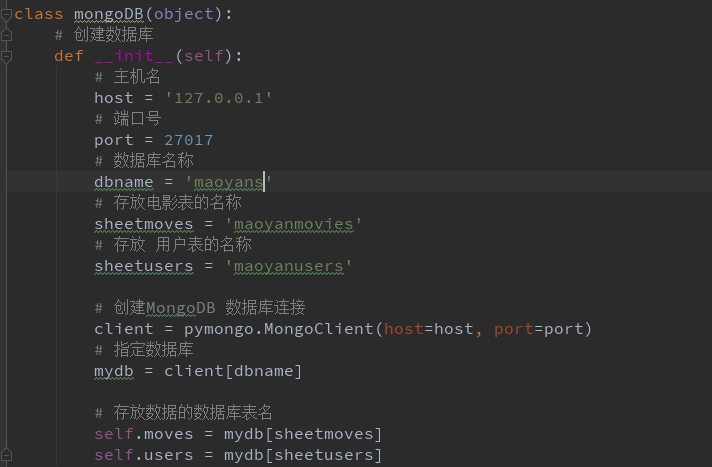


图 5 数据库创建

评论

user

N

\_id

comment

time

score

film

N

Film\_name

country

\_id

Box\_office

Comment\_people

score

length

released

name

图 6 数据库的ER图

在创建用户集合、电影集合的时候，还需要添加一个两者关系的集合。

用户集合：

{

\_id:”u1”,

name:”小明”

}

电影集合：

{

\_id:”f1”,

name:”影”,

country:”中国”,

score:’9分’,

......

}

关系集合（用户u1评论了电影u2）：

{

\_id:1,

users\_id:’u1’,

films\_id:’f1’,

time:2018-11-14,

score:’8分’,

Comment:”这部电影看着很感人，很有代入感！”

}

用户集合（users）字段格式

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| \_id | String | 用户主键 |
| name | String | 用户姓名 |

电影集合（film）字段格式

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| \_id | String | 电影主键 |
| name | String | 电影名字 |
| type | String | 电影类型 |
| Length | String | 电影时长 |
| Released | String | 电影发布 |
| Score | String | 电影评分 |
| People | Int32 | 电影评论人数 |
| Box\_office | Int32 | 电影票房 |

关系集合（connection）字段格式

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| \_id | String | 关系主键 |
| User\_id | String | 用户主键 |
| Film\_id | String | 电影主键 |
| Time | String | 电影评论时间 |
| Score | String | 电影评分 |
| Comment | String | 电影评论 |

**4.2 服务器搭建**

为了让用户能够用过浏览器查看数据，该系统搭建了一个服务器，将用户浏览器发来的查询请求进行处理，然后服务器从数据库中读取数据；将数据返回给前端。前端将服务器给的数据进行可视化处理。

服务器的搭建该系统采用了Express服务器框架，它是是一种保持最低程度规模的灵活 Node.js Web 应用程序框架，为 Web 和移动应用程序提供一组强大的功能。

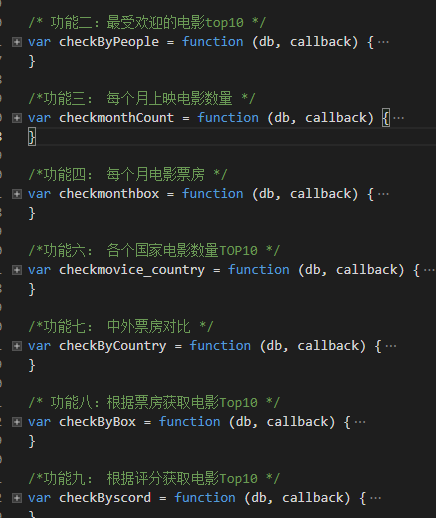


图 7 服务器端查询模块

**4.3 电影数据展示**

首先浏览器向服务器发送查询数据请求，服务器进行响应，通过json将数据传输给index.js 页面，然后inde.js 调用各种模块去解析数据，最后显示到页面上。



图 8 前端解析数据

**4.4 电影推荐**



**5 使用过程**

5.1 启动服务器

打开一个终端输入npm start 命令，然后服务器启动，打印提示信息，服务器启动成功，链接数据库成功。

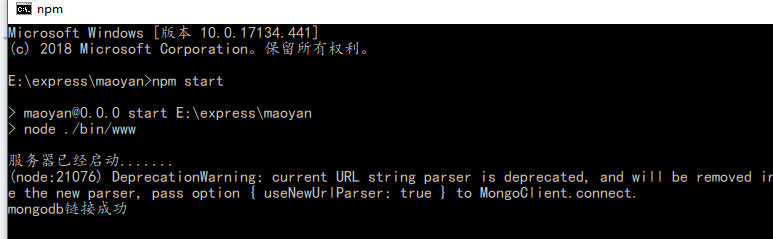


图 9 启动服务器

启动nginx 程序，将本地服务器地址反向代理，让用户可以输入www.movie.com 就可以看到电影相关数据，而不是输入Ip地址+端口号。

5.2 打开浏览器输入地址 [www.movie.com](http://www.movie.com) ，显示数据

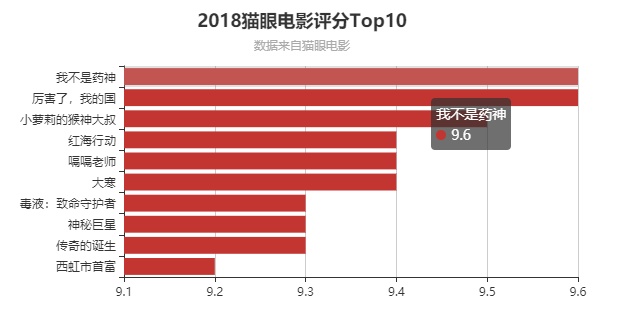


图 10 电影评分top10

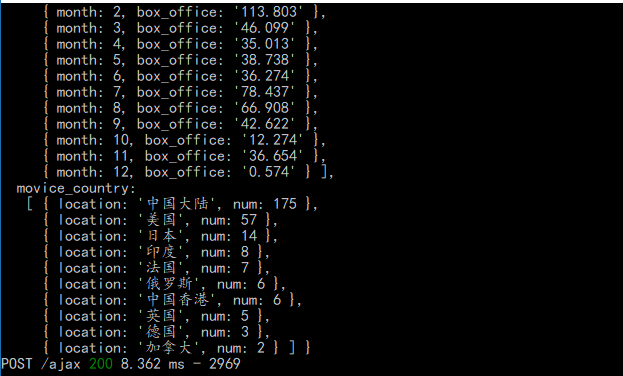


图 11 服务器处理请求

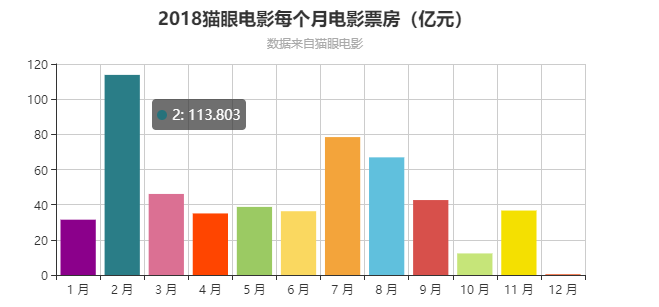


图 12每月电影票房

这里我们就可以发现，2月份也就是春节档电影票房最多，其次就是7、8月份暑假期间。



图 13 电影类型分布

将2018年上映的所有电影类型进行统计，我们可以得出剧情、喜剧类型的电影最多。

5.3 电影推荐

用户输入电影名称，向服务器发送电影推荐请求，服务器调用脚本执行机器学习的KNN推荐算法，该算法根据用户输入电影的类型进行分类，然后向用户推荐评分最高的5部电影。

