## 实验环境以及技术选型

#### 实验环境：

操作系统: windows 10  
数据库管理系统: SQL server 2018 DEV  
系统设计与开发用到的工具：visual studio 2017

#### C#的窗体程序开发

本系统设计的主题采用了C#的TreeView 组件实现不同界面的导航菜单。

#### SQL server数据库

#### Python获取电影数据

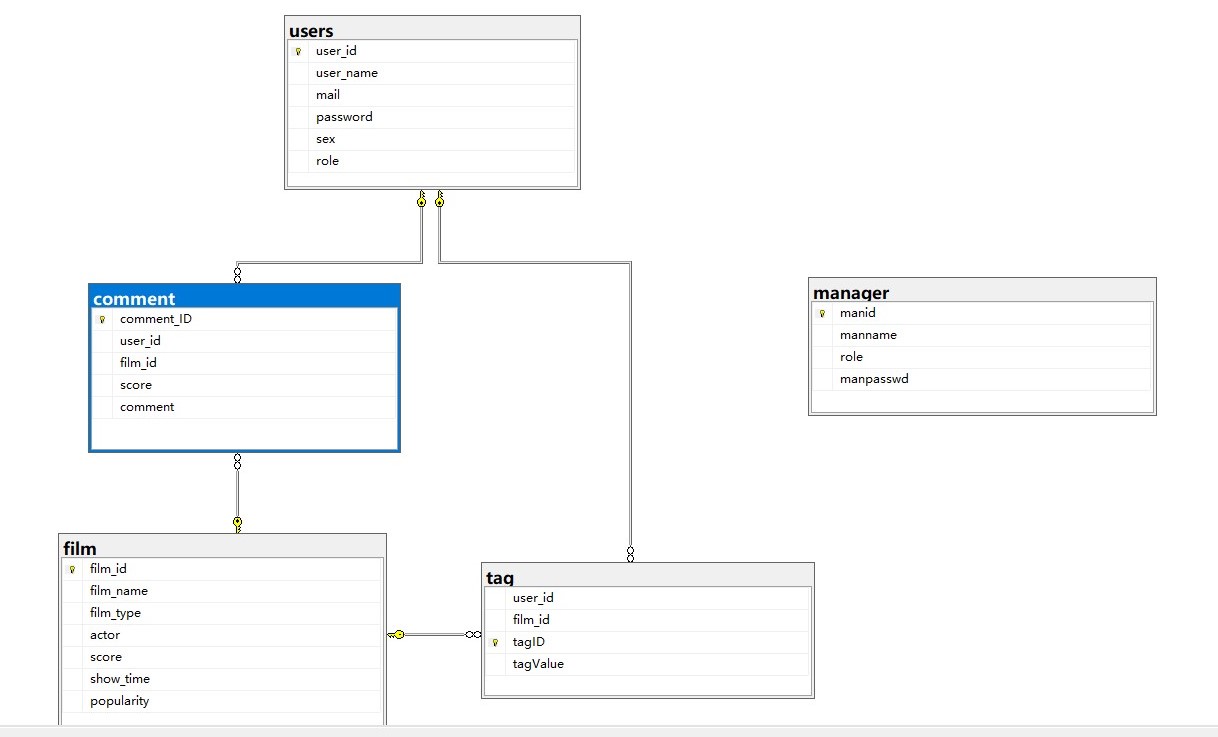
使用python的request、json解析的库，解析猫眼电影网页，获得排名榜前100的电影资讯。

## 需求分析

* 本系统包括两类使用人员：用户、管理员。
* 管理员可以使用的功能：修改用户信息、对电影进行增删查改，以及针对某些用户的不良言论进行删除。
* 用户可以使用的功能：搜索电影（输入电影全名进行精确搜索、根据输入电影名称中的关键字或者关键词进行定位、选取电影类型和设定评分阈值进行模糊搜索、根据评论热度推荐电影）、对已看过的电影进行评论、操作已收藏的电影、根据个人喜好对电影进行相关操作。
* 管理员和用户都拥有基本的注册、登录、修改密码的三个功能。

## 4、数据库设计

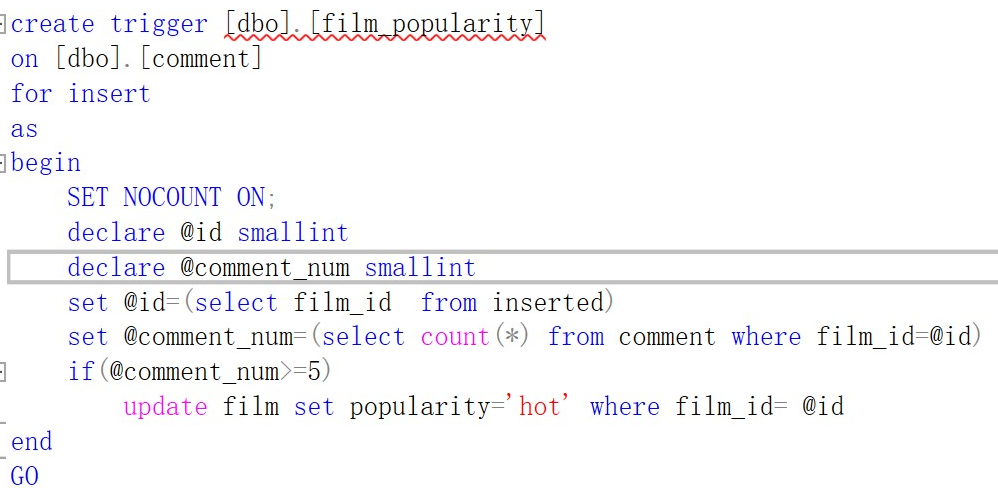
### 4.1数据库关系图



### 4.2设计触发器

###### 触发器[film\_popularity]：

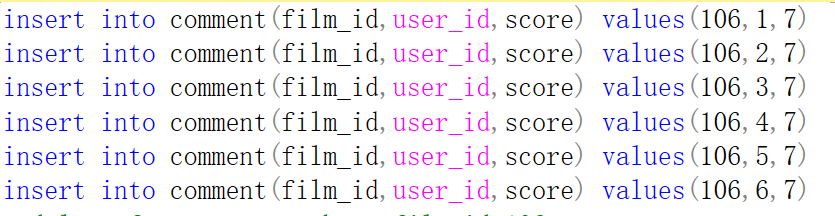
* + 功能：某电影中的评论数量达到一定数量后，该电影的热度属性会被置为”hot”，随后当用户查看评论热度榜单时，可以看到该电影。



* + 运行结果：当多于5个用户对ID为 的电影分别进行评论后，comment表上的触发器检查到insert操作之后，评论数量达到5，因此更新电影表中该电影的评论热度
    - 更新评论热度之前的榜单



* + - 方便起见，直接在SQL server中构造插入语句，模拟不同用户对ID=106的电影进行评论的行为

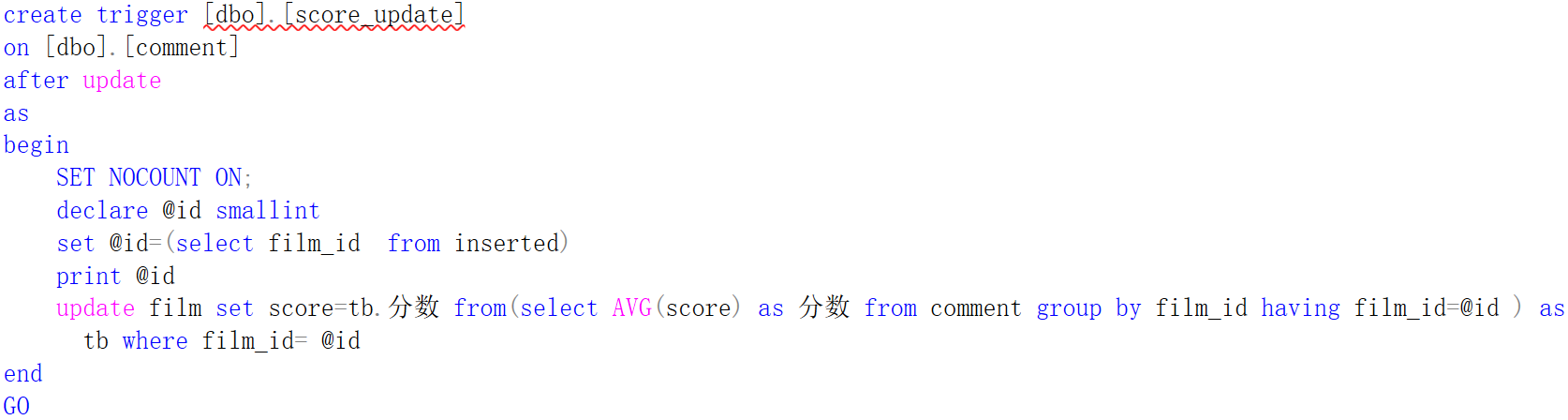


* + - 由于评论数量超过5，因此触发器被激活，评论热度榜单即显示出该电影



###### 触发器[score\_update]：

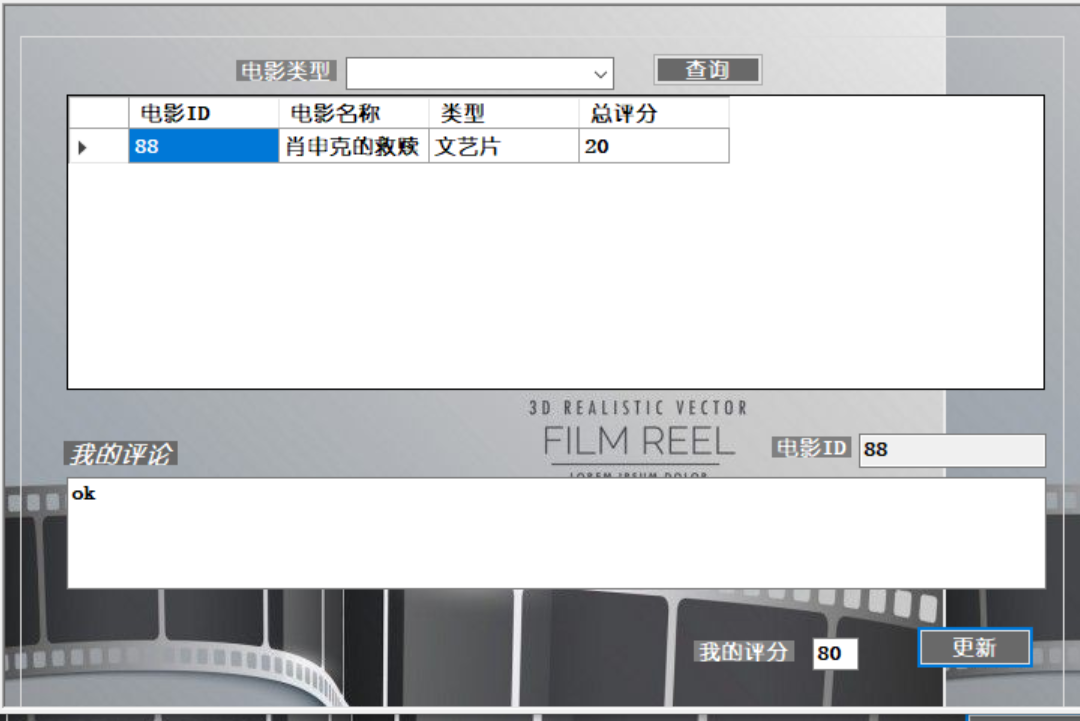
* + 功能：对电影的评分进行实时更新。由于每部电影的评分为所有用户对其评分的均值，因此，每当有用户对某电影进行评分或更新评分之后，该电影的评分就需要进行更新。



* + 运行结果:ID=88的电影原先评分为24，某用户对其评分之后，comment表的触发器检测到update操作，因此更新电影表中该电影的评分



更新后，评分为20



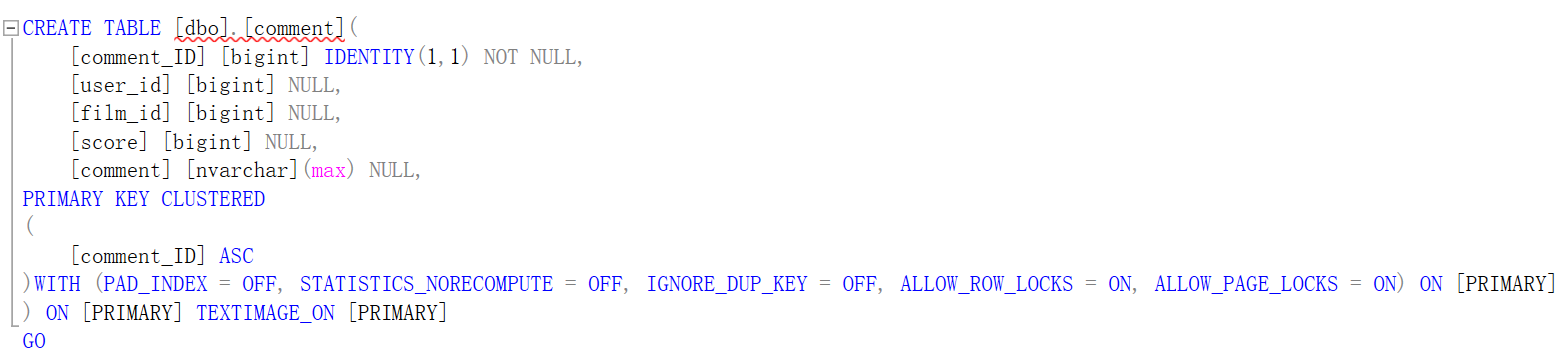
### 4.3 数据库逻辑及物理结构设计与实现

（在此仅列举部分建表过程）

#### 建立数据库



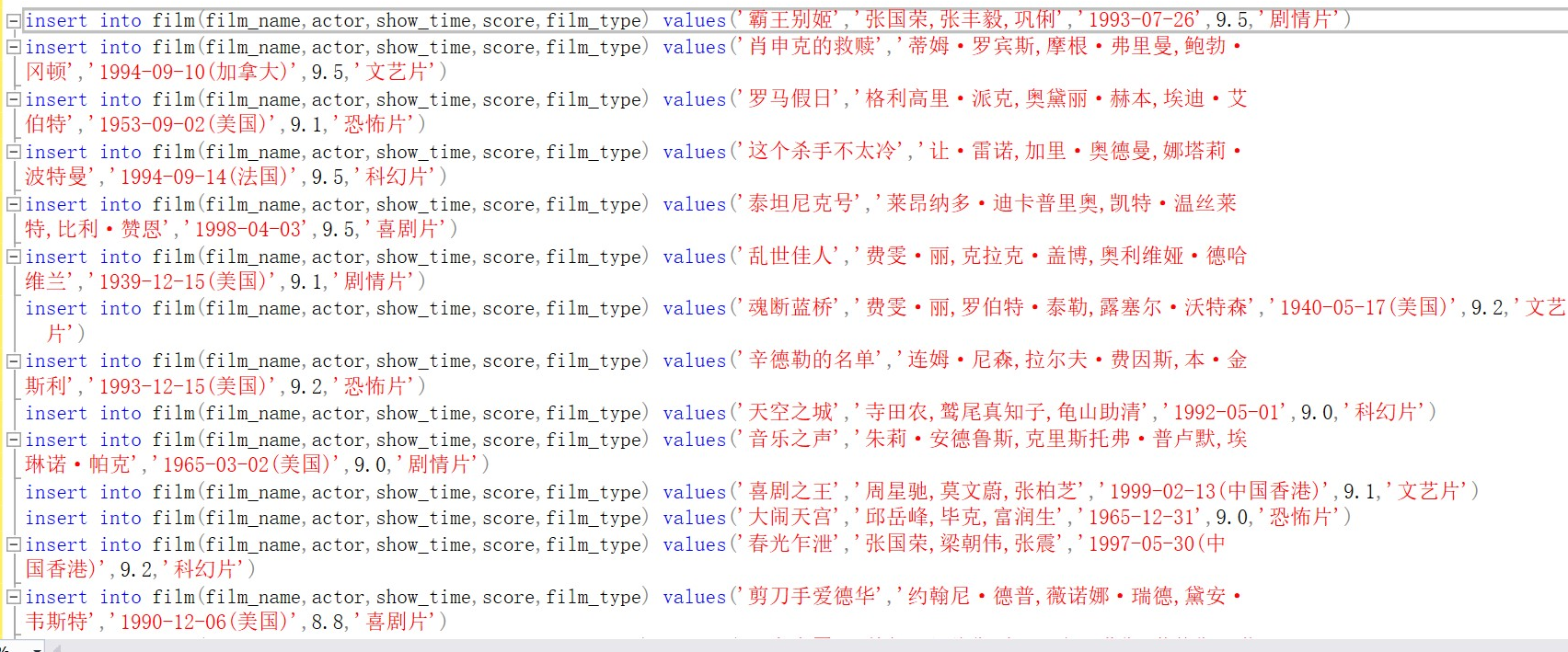
#### 建立表comment



#### 建立comment 表中的外键，关联到film表，同时建立更新以及删除规则皆为CASCADE



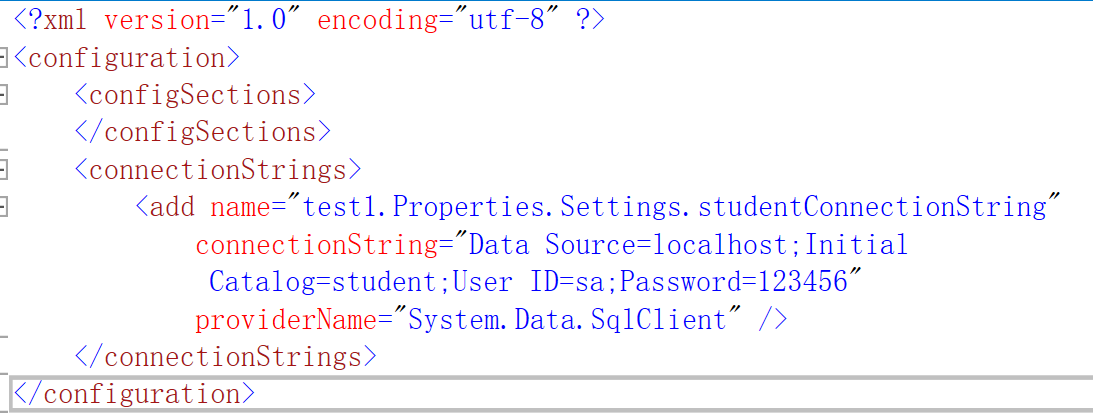
#### 数据的录入（先使用python爬取网站上的电影信息，之后构造sql语句，向数据库中插入数据）



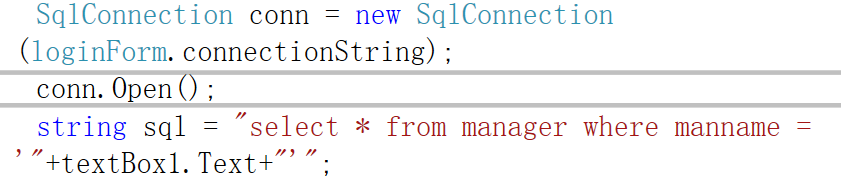
## 5、关键技术介绍以及功能展示

### 数据库的连接

###### 在C#项目配置文件app.config中定义数据库的连接字符串



###### 在每次需要与数据库交互之前，构造SqlConnection的对象，调用配置文件（或登录页面）中的定义的数据库连接字符串，即可连接

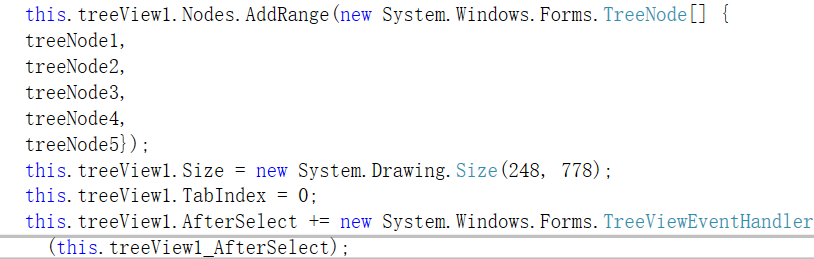


### 6.1使用treeview实现页面导航菜单

#### 注册treeview节点信息



###### 规定节点在导航栏中显示的顺序



###### 定义点击节点之后的事件处理函数（跳转至相应页面）



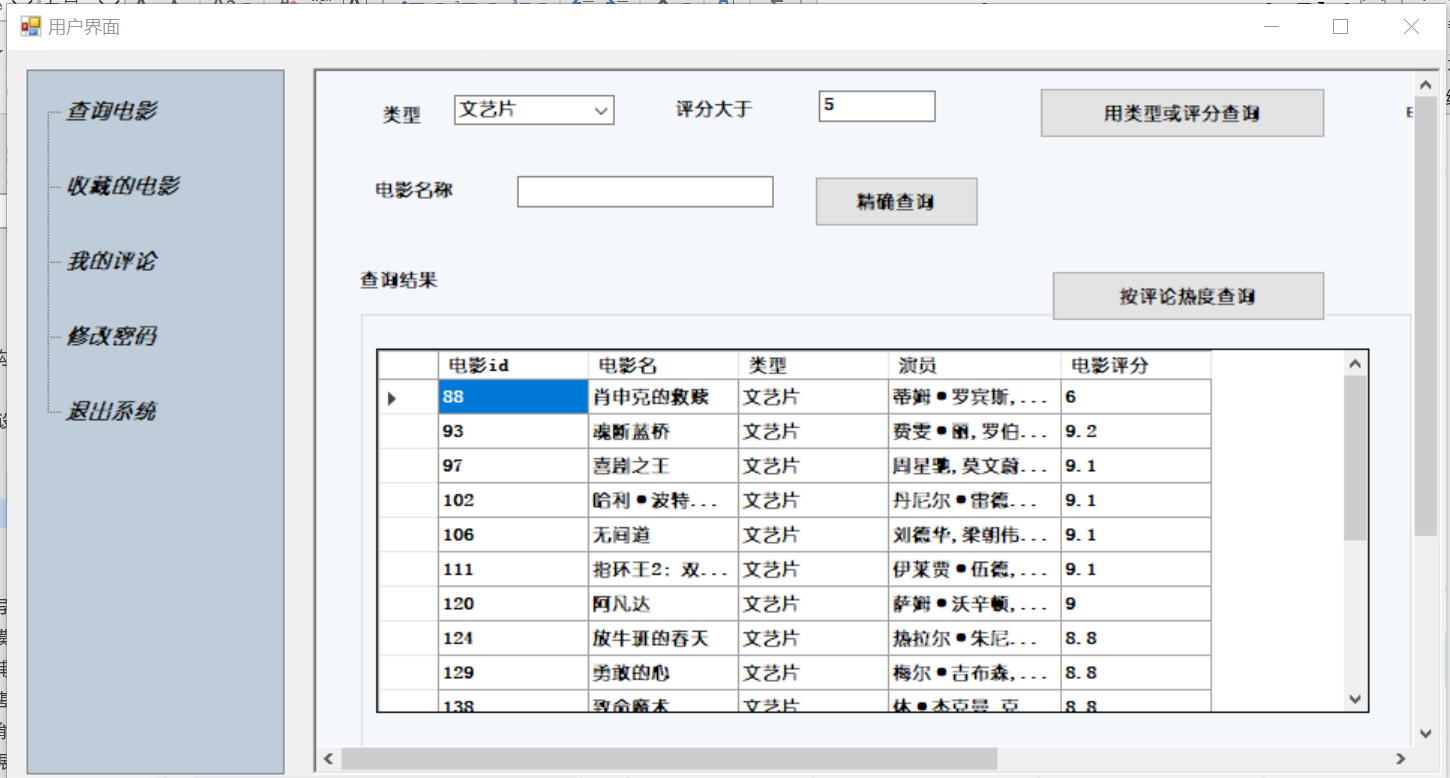
### 6.2实现根据电影关键词的模糊搜索

此处用到C#语言的SqlDataAdapter对象，将已经构造好的sql语句和连接字符串创建SqlDataAdapter实例，提交执行后，得到一个虚表(dataset对象)存放返回的结果，之后的操作都将在此虚表中进行。



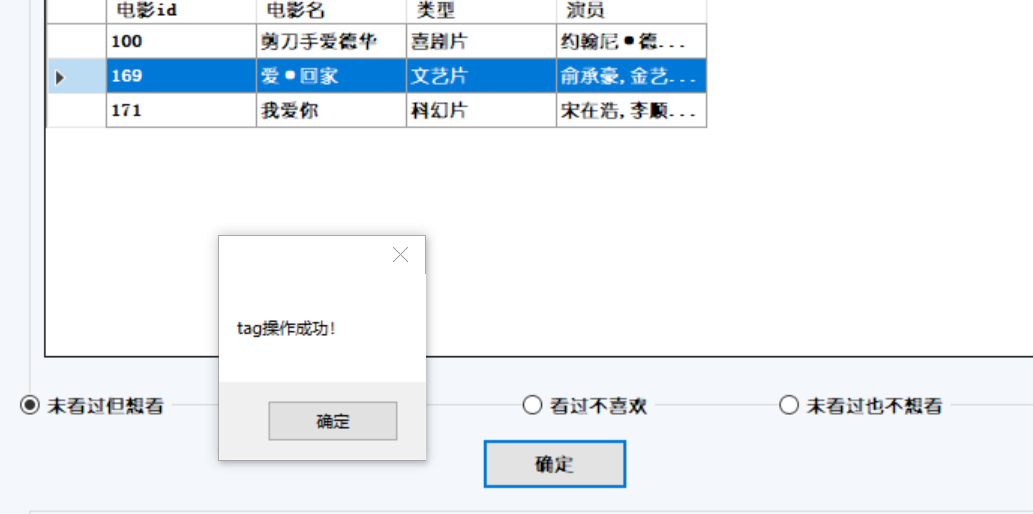
通过ds.Tables[0].Rows[i][1].ToString() 定位到虚表中的某行某列，可以得到已有的电影名称

* 基于类型或者评分查询



* 输入关键词”爱”,即可搜到包含关键词的电影，同时操作

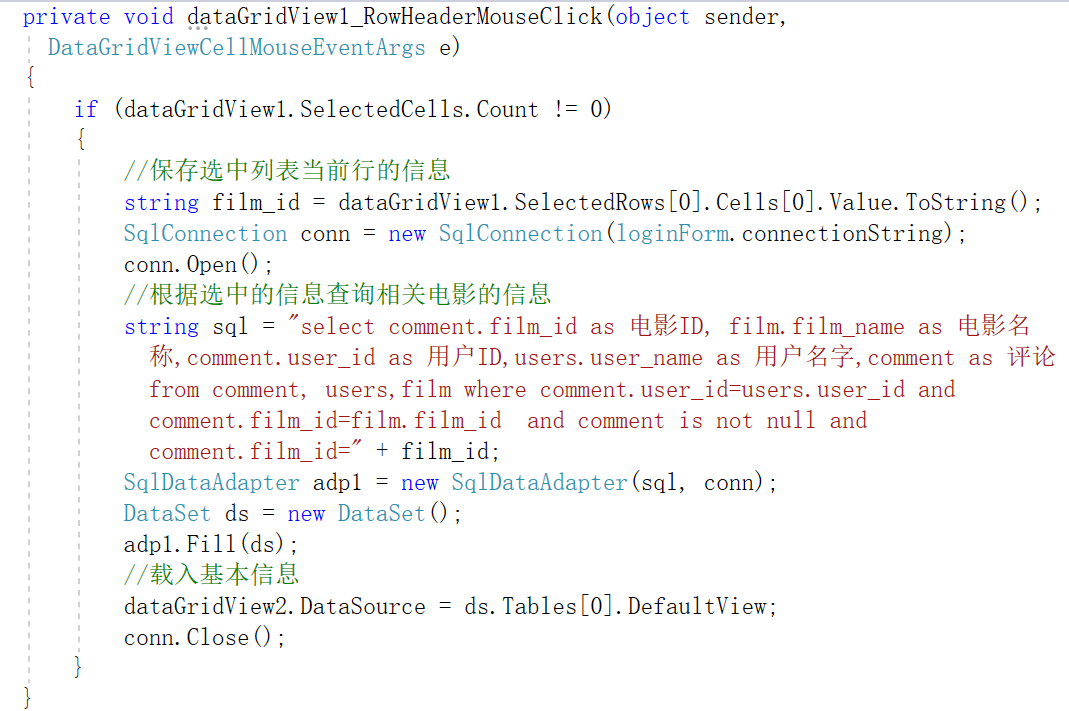




* 点击查询结果中的电影即可查看相关评论



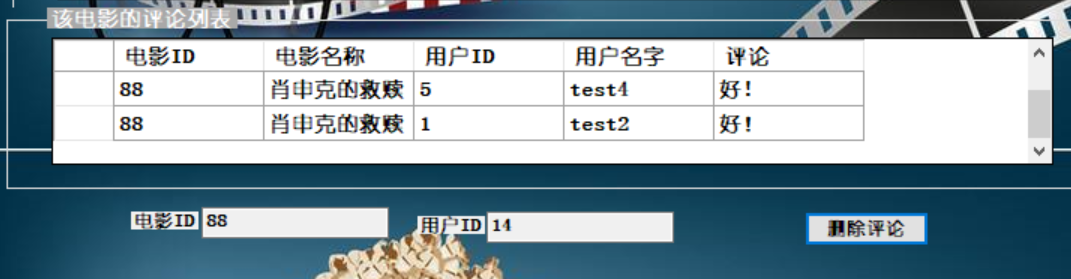
### 6.3 管理员根据选中电影，展示对应的评论



* 查看某电影的所有评论，可以删除某用户评论



* 该评论即从评论列表中移除



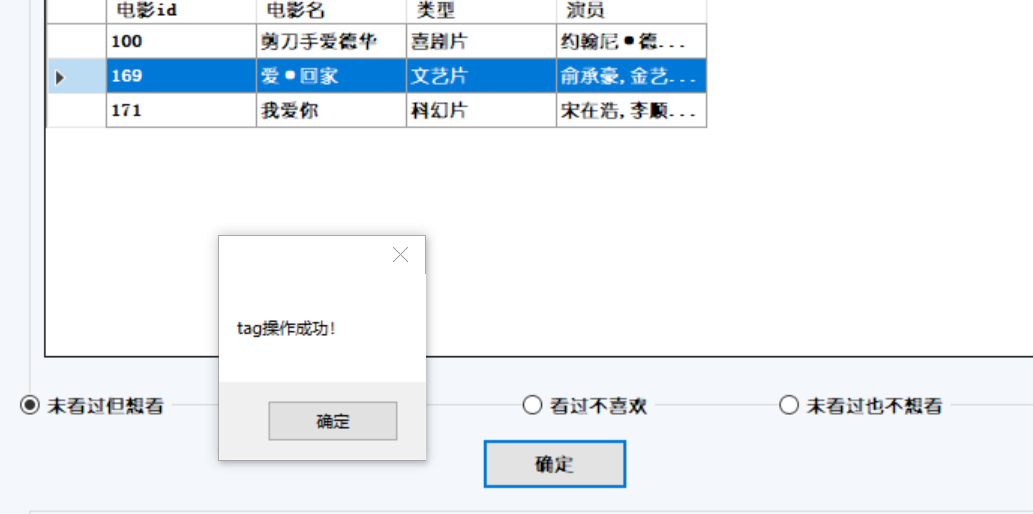
### 6.4 用户给电影打标签时，基于Tag表已有的值对于操作进行处理

此处用到SqlCommand对象封装的函数ExecuteScalar()，返回查询执行结果集的第一行第一列

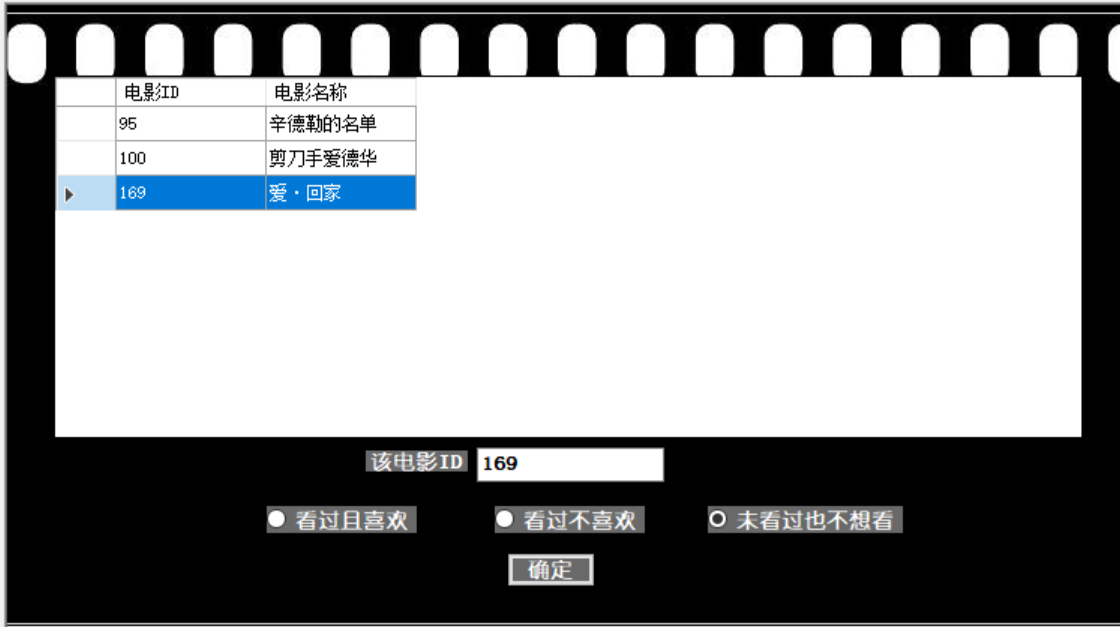


Flag=0即尚未给该电影打过标签

* 对于某电影贴上”未看过但想看”的标签

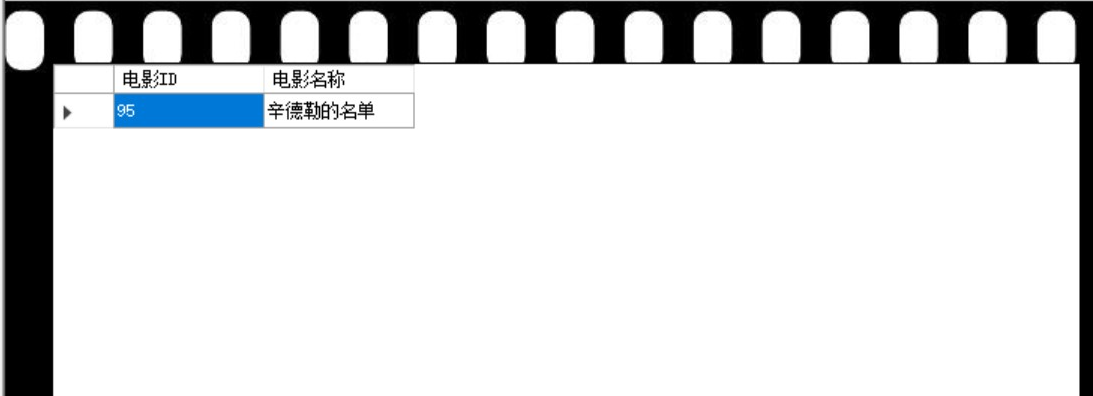


* 之后可在”我的收藏”页面中找到该电影



* 若更改某电影的标签，则该电影会从收藏列表中被移除





## 6. 项目总结

本次小组合作完成电影管理系统的过程让我收获颇丰。

首先，作为组长，我深刻体会到了领导小组的不易和小组合作的重要性。从项目选题的构思，到数据库的设计优化，再到项目的实现，我需要不断考虑到小组成员技能上的不同以及项目进度的发展，不断调整分配给组员的任务。同时，组员之间的合作交流、大家发挥各自的专长和观点上的碰撞，让整个项目的开发充满了乐趣。

其次，作为本管理系统开发的主要编程人员，我学习到了很多关于使用C#进行系统开发的技能。通过参考网络上的教程，我从基础部分开始学习项目的配置、构建和调试方法，重新复习面向对象编程的相关知识，并将其实践，大大加深了我对面向对象的理解，领略其编程思想的精髓。

最重要的是，通过此项目的开发，我深刻体会到数据库设计的重要性以及理解数据库设计的理论知识。在项目的起步阶段，我们需要设计符合实际功能的数据库，并且尽可能优化表的结构。在此过程中，我们将课本上所学的范式规则、外键、约束条件、保持表的完整性等性质应用到表的设计中，学会了灵活运用它们，并在不断地尝试、碰壁中更深刻理解相关知识。之后，在编程阶段，以嵌入式SQL语句实现和后台数据库交互时，我逐步熟悉了增删查改语句的格式，以及异常情况的处理。最后，通过设计两个触发器和调试的过程，我才真正理解了触发器的本质和优点。但是，由于现阶段知识和能力的不足，我们尚未运用视图、优化表结构，建立索引、查询优化和事务执行的相关内容。今后，我们将继续尽力完善该系统，以实现更多功能。

## 7.参考文献

* 博文：触发器deleted 表和 inserted 表详解

<https://www.cnblogs.com/joetao/articles/2250521.html>

* SQL Server 2008 [SQL.Server2008中文版关系数据库基础与实践教程]
* 《Pro C# 5.0 and the .NET 4.5 Framework》