# 《数据库系统原理》课程设计

# 系统设计报告

# 

# 题目：电影购票系统

# 

# 

# 

# 

# 

## 学号及姓名：

## 1406XXXX

## 1406XXXX

2017 年 1 月 5日

|  |  |  |
| --- | --- | --- |
| 组内任务说明 | | |
| 姓名 | XXX | XXX |
| 系统设计 | 数据流图分析  ER图分析  逻辑模式分析及规范化 | 用例图分析  ER图分析  逻辑模式分析及规范化 |
| 系统实现 | 前端  后端  数据库 | 前端  后端  数据库 |
| 文档撰写 | 数据库实现报告 | 数据库设计报告 |

## 一、系统概述

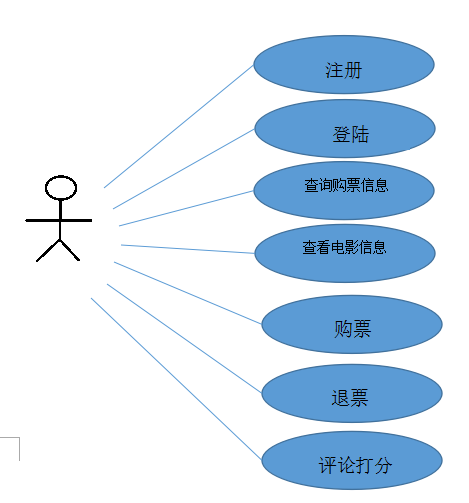
本文档是电影购票系统的设计文档，该系统的基本特性是能够满足管理员对数据库中关于电影的一些信息的增添以及删除，以及用户的查询购票操作。本购票系统将角色分成两类，用户和管理员：提供用户注册、电影信息查询、电影放映安排、电影票购票和退票等功能。

## 二、系统需求分析

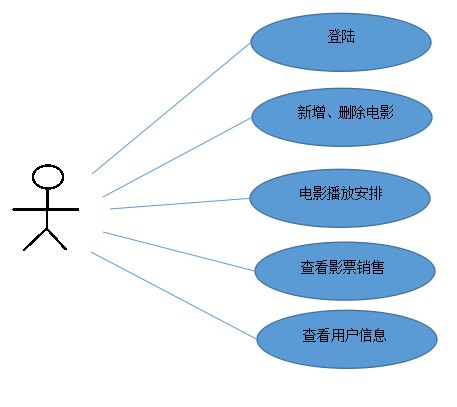
以用例图的形式给出系统功能需求的分解结构，并对用例模型中的参与者和用例进行详细的描述。同时用数据流图展示功能需求和数据需求的联系以及数据流。

### 2.1用例图

用户用例图：



管理员用例图：



### 2.2针对两个用例的需求分析：

2.2.1用户需求：

1.用户实现基础的注册、登陆。

2.能够查询电影上映情况，并选择当天上映或即将上映且有余票的电影进行购票。

3.对未过期的影票，可以实现退票功能。

4.可以查询自己的未观看的和已购买的、已退票的电影信息

5.可以查询电影的基本信息(导演、主演、宣传片等)。

6.对网站上公布的电影，用户可以进行评分和评论，并随时查看排名前 10 的电影。另外用户可以对某部电影进行评论。

2.2.2管理员需求：

1. 能登陆管理员账号。
2. 能在系统中新增、删除一部电影。
3. 能选择电影进行安排(放映时间、放映影厅等)。
4. 管理员能随时查看影票的销售情况、用户注册数等。

### 2.3用例详解

1. 用例名称：用户登陆

用例描述：根据用户输入的用户名和密码判断用户的身份。

前置条件：无

后置条件：判断用户账号是否存在或者是否匹配密码，如果符合，登陆。

基本操作流程： 以用户身份进入相应的界面。

可选流程：如果密码不正确，提示重新输入密码；如果用户名不正确，提示没有此用户。

2.用例名称：管理员登陆

用例描述：根据管理员输入的用户名和密码判断管理员身份。

前置条件：无

后置条件：判断管理员账号是否存在或者是否匹配密码，如果符合，登陆。

基本操作流程： 以管理员身份进入相应的界面。

可选流程：如果密码不正确，提示重新输入密码； 如果管理员名不正确，提示没有此管理员。

2.用例名称：查询电影信息

用例描述：由用户进行操作，查询电影信息，可以显示电影上映情况、导演、主演、宣传片等信息。

前置条件：登陆

后置条件：无

基本流程： 显示电影的信息。

3.用例名称：购票

用例描述： 用户购买一张电影票。

前置条件：用户身份登陆。

后置条件：选择的场次有余票且没上映。

基本流程：提示购票成功，并且购票信息增加一个记录。

可选流程：如果该场次电影无法购买，提示无法购买。

4.用例名称：退票

用例描述： 用户退订一张电影票。

前置条件：用户身份登陆。

后置条件：该场次未上映。

基本流程：提示退票成功，并且购票信息删除一个记录，退票信息增添一个记录。

5.用例名称：查询购票信息

用例描述：由用户进行操作，查询自己购买电影票的信息。

前置条件：用户登陆。

后置条件：无

基本流程： 显示自身购票信息，退票信息。

6.用例名称：评论打分

用例描述： 用户对一部电影打分。

前置条件：用户身份登陆。

后置条件：无

基本流程：选择一部电影进行打分，增加一个评论信息。选择一部电影打分，系统会自动根据所有用户的打分给出所有电影的评分排行榜。

7.用例名称：新增或删除电影

用例描述：由管理员新增或删除一部电影的信息。

前置条件：管理员身份登陆。

后置条件：无

基本操作流程：新增电影，就新增一条电影的记录；删除就是删掉该电影的记录。

8.用例名称：电影播放安排

用例描述：由管理员安排一个场次的电影播放。

前置条件：以管理员身份登陆。

后置条件：安排的时间无冲突。

基本操作流程：新增一个电影播放场次或者删除一个场次。

9.用例名称：查看影票销售

用例描述：管理员查看某场次的售票情况。

前置条件：以管理员身份登陆。

后置条件：无

基本操作流程：查看电影售票数等。

10.用例名称：查看用户信息

用例描述：管理员查看用户相关信息。

前置条件：以管理员身份登陆。

后置条件：无

基本操作流程：显示用户注册数、用户电影的关注量等。

11.用例名称：注册

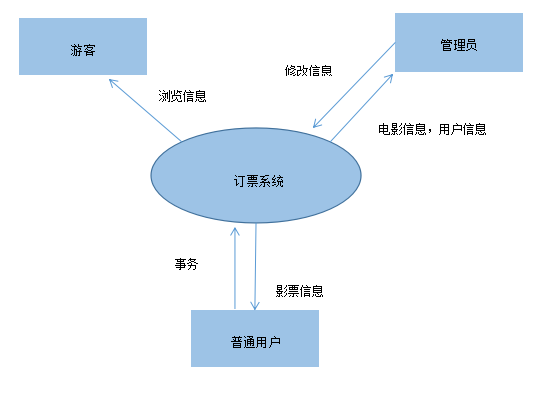
用例描述：新用户注册账号

前置条件：无

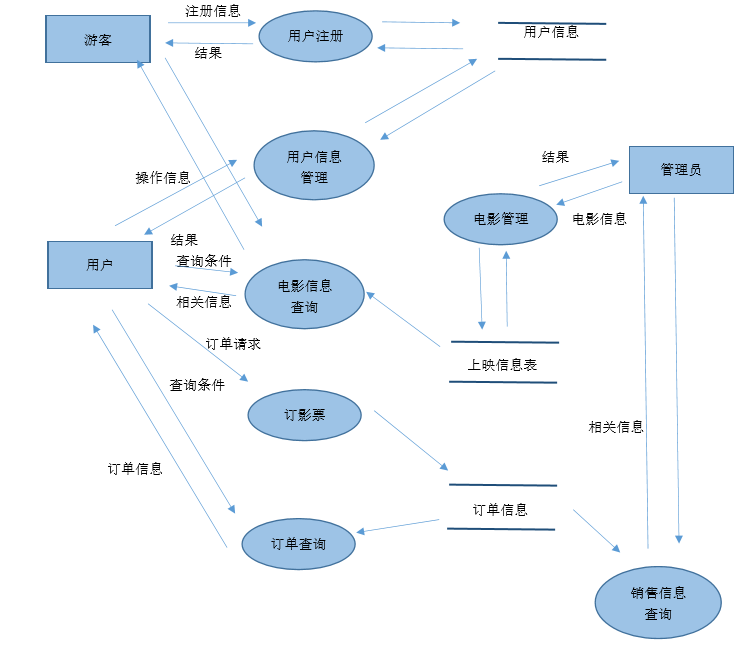
后置条件：账号名不重复

基本流程：输入邮箱、昵称、密码完成注册。

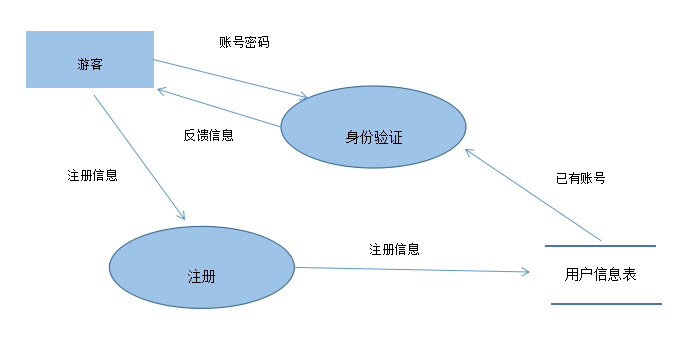
### 2.4数据流图：



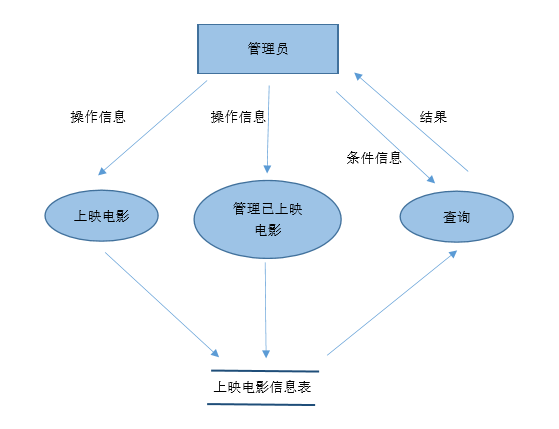
0层数据流图



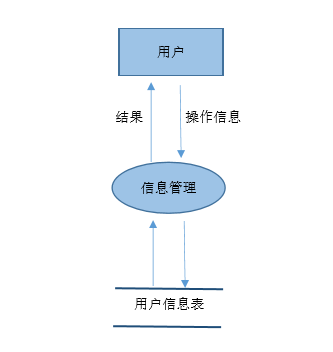
1层数据流图 订票系统



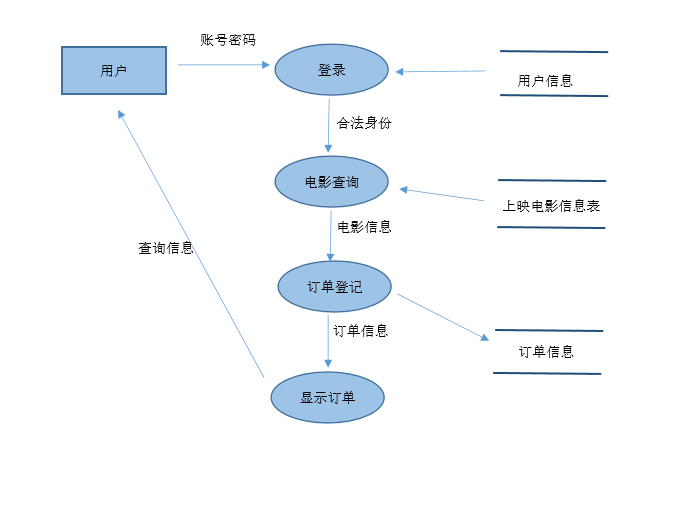
2层数据流图 用户注册



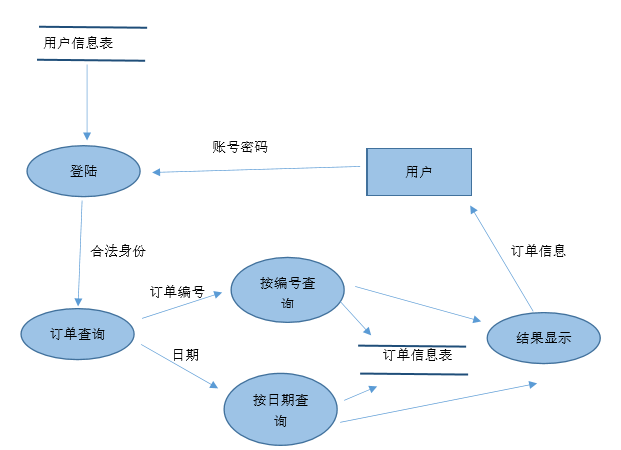
2层数据流图 电影管理



2层数据流图 用户信息管理



2层数据流图 购票

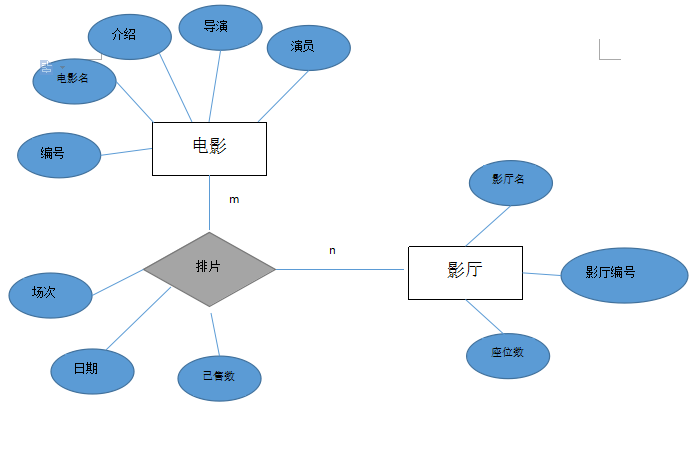
 2层数据流图 购票信息查询

**2.5 数据元素表**

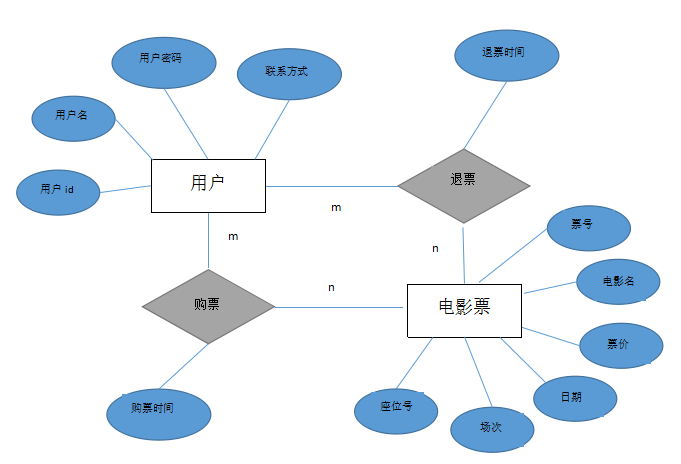
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据元素名 | 数据类型 | 数据宽度 | 小数位数 | 单位 | 值约束 | 允许空值 |
| 电影编号 | int | 10 | 0 | 无 | 大于0 | 否 |
| 电影类型 | string | 10 | 0 | 无 | 无 | 否 |
| 电影名称 | string | 20 | 0 | 无 | 无 | 否 |
| 影厅名 | string | 20 | 0 | 无 | 无 | 否 |
| 影厅编号 | int | 10 | 0 | 无 | 大于0 | 否 |
| 座位数 | int | 3 | 0 | 无 | 大于0 | 否 |
| 管理员id | int | 10 | 0 | 无 | 无 | 否 |
| 管理员名 | string | 20 | 0 | 无 | 无 | 否 |
| 管理员密码 | string | 20 | 0 | 无 | 无 | 否 |
| 用户联系 | string | 20 | 0 | 无 | 无 | 否 |
| 用户名 | string | 20 | 0 | 无 | 无 | 否 |
| 用户密码 | string | 20 | 0 | 无 | 无 | 否 |
| 用户ID | int | 10 | 0 | 无 | 大于0 | 否 |
| 排片电影名 | string | 20 | 0 | 无 | 无 | 否 |
| 排片日期 | date | 10 | 0 | 无 | 合理 | 否 |
| 排片场次 | date | 5 | 0 | 无 | 合理 | 否 |
| 已售票数 | int | 3 | 0 | 无 | 大于0 | 否 |
| 评论编号 | int | 10 | 0 | 无 | 大于0 | 否 |
| 评论内容 | string | 1000 | 0 | 无 | 无 | 是 |
| 评论电影名 | string | 20 | 0 | 无 | 无 | 否 |
| 票号 | string | 10 | 0 | 无 | 无 | 否 |
| 购票用户ID | int | 10 | 0 | 无 | 大于0 | 否 |
| 购票时间 | date | 20 | 0 | 无 | 合理 | 否 |
| 退票时间 | date | 20 | 0 | 无 | 合理 | 否 |
| 票价 | int | 3 | 0 | 无 | 大于0 | 否 |
| 座位号 | int | 3 | 0 | 无 | 大于0 | 否 |

## 数据库系统的概念模式

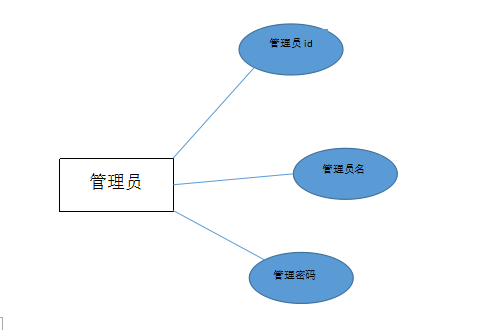
## ER图：



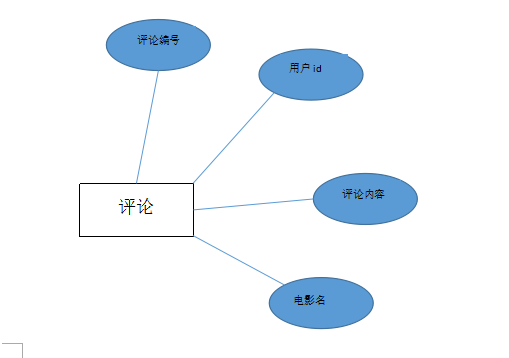
排片ER图（电影属性过多，所以有些属性没有画出来）



购票退票ER图



管理员ER图



评论ER图

## 四、数据库逻辑模式设计

### 4.1 数据库关系模式

电影（电影编号，电影名, 类型,短介绍,长介绍，导演，演员，收藏人数，评分人数，评价人数，得分，添加日期，票价）

管理员（管理员 id，管理员名，管理员密码）

影厅（影厅编号，影厅名，座位数）

排片（电影名，影厅编号，日期，场次，已售数）

用户（用户 id，用户名， 用户密码, 联系方式 ）

评论（评论编号，电影名，评价内容，用户 id）

电影票（票号，电影名，票价，日期，场次, 座位号）

购票（用户 id，票号，购票时间）

退票（用户 id，票号，退票时间）

### 4.2 关系模式范式等级的判定与规范化

规范到 3NF

1）1NF：

经过检验，所有关系模式的所有属性均为不可分的数据项，所以

均属于 1NF。

2）2NF：

可以发现，对于每个关系模式，在 1NF的基础上每个非主属性均完

全函数依赖于任何候选码，所以均属于 2NF。

3）3NF：

由于所有关系模式均属于 2NF，而且每个关系模式的每个非主属

性都不传递依赖于此关系模式的任何码，所以所有关系模式均属于

3NF。

### 4.3 数据库设计优化

1）确定数据依赖，对关系模式进行逐一分析

在前面的步骤中，已经对所有关系模式进行分析，并规范到 3NF。

2）检查对于特定的应用环境，以上关系模式是否合适。

经过分析我们认为，随着系统的使用，将会产生大量的影票、购

票、退票信息，而为了检索用户的购退信息而做的连表操作显然是必

须的。但当数据量很大的时候连表操作将耗费大量资源，所以以后如果选择优化，可以将购退票与电影票关系合并。

电影票（票号，电影名，影院名，日期，场次，购票时间，退票时

间，用户名）

合并后仍属于 3NF。