CS425 In Class Project document

Online Ticketing and Theatre Management System

Design & Implement

by

your name

your name

your name

§ Document Info

This page handled the document modifying history and specified the members who attend to writing and design.

Change Log

| Version | description | members | date |
| --- | --- | --- | --- |
| 0.1 | initial document frame created, given out basic content | Jinyang Li | 10/10/2015 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Feature Ownership

| Feature | Tables In SQL | Owner |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

§ Content Table

1 Introduction

1.1 Organization of document

1.2 Requirements of project

1.3 ER-Diagrams & SQL Schemas\*

2 Concept Design

2.1 Data entity description

2.2 Entity relation description

3 Implement

3.1 Physical structure

3.2 Safety

3.2 Table implement structure

\*These content were in the end of this document as the homework indicated.

1 Introduction

1.1 Organization of Document

The document is oriented to explain a database design which was not associated with any software systems, and the database meets the requirements indicated in homework-03 instruction docs.

In this document, the two important parts were including:

- ER-diagrams and SQL schemas

- Database design explanation

All access points of above important content can be found in the table of content.

(考虑在这里放上目录的详细解释 看到时候空间够不够)

1.2 Requirements of project

The name of this project is

Online Ticketing and Theatre Management System

This project require:

- Oracle database driver or any database driver which support Oracle database.

- unknown software system using this database.

1.3 ER-Diagrames & SQL-Schemas

In the end of the document.

2 Design

2.1 Data Entity Description

\*Pk means primary key in table. This section given out the 3NF of database schema.

After analyzing the requirement, the x个 tables needed for this project.

| filed\_name | data type | PK | nullable | description |
| --- | --- | --- | --- | --- |
| ID | int | yes | not | unique identify number |
| NAME | varchar(50) |  | not | the name of theatre |
| location | varchar(100) |  | not | the location of theatre |

Theatre table上面是正统的SQL开发文档用到的格式3NF 但是我们作业好像篇幅有限制,好像不够

Movie table

| filed\_name | data type | PK | nullable | description |
| --- | --- | --- | --- | --- |
| ID | int | yes | not | unique identify number |
| title | varchar(50) |  | not | the name of theatre |
| director | varchar(100) |  | not |  |
| star | int |  | yes |  |

Movie\_type table

| filed\_name | data type | PK | nullable | description |
| --- | --- | --- | --- | --- |
| ID | int | yes | not | unique identify number |
| type | varchar(50) |  | not | the name of type, with constraint |
| description | varchar(100) |  | not |  |
| MID | int |  | yes | foreign key refer to ID in movie table |

….妈的好多啊。。。剩下的交给你们了我去做schema 做成hw1那种然后你们来填这个

2.2 Entity relation description

考虑删除这个part太占篇幅

3 Implement

3.1 Physical structure

These content was in the end of page as part of SQL Schema.

这里就是具体的sql语句 比如create table xxx

3.2 Safety

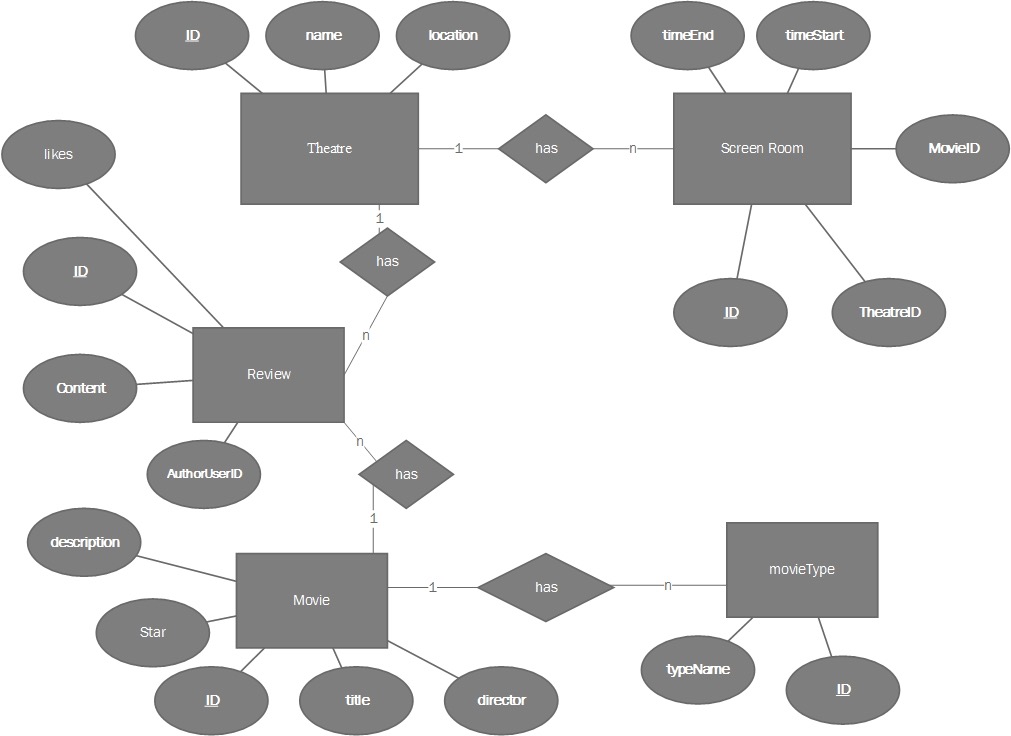
用户权限设置，先把前面撸了再来弄这个吧。。

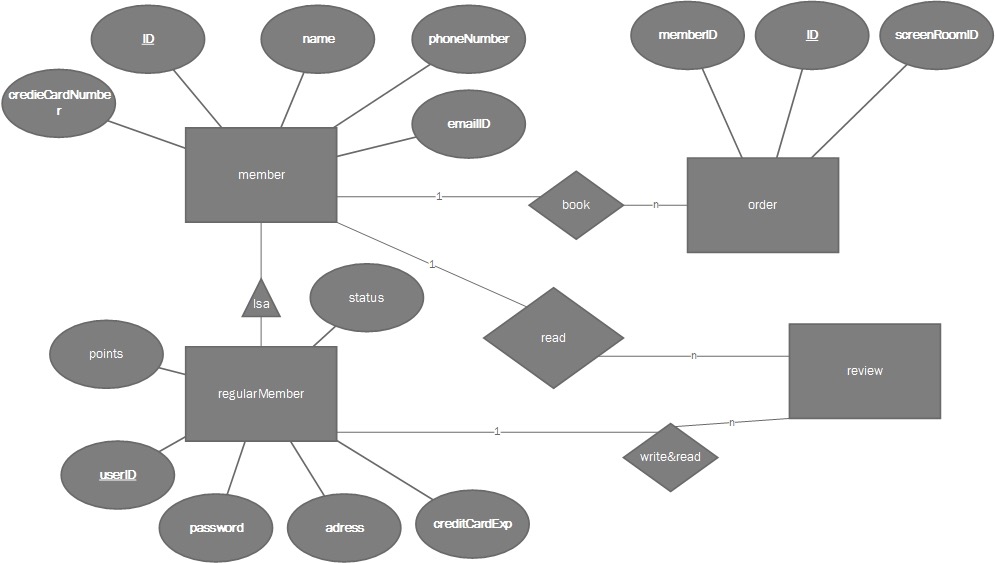
3.3 Table structure

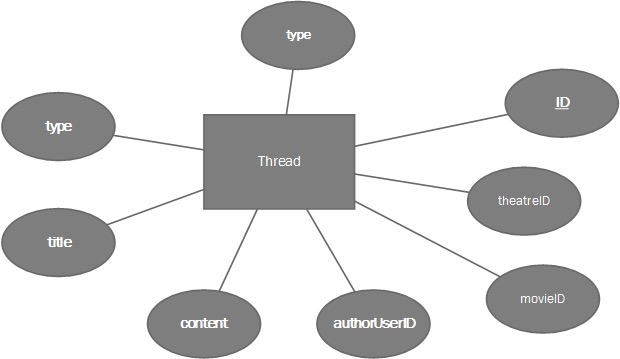
看篇幅删不删这个section。。。

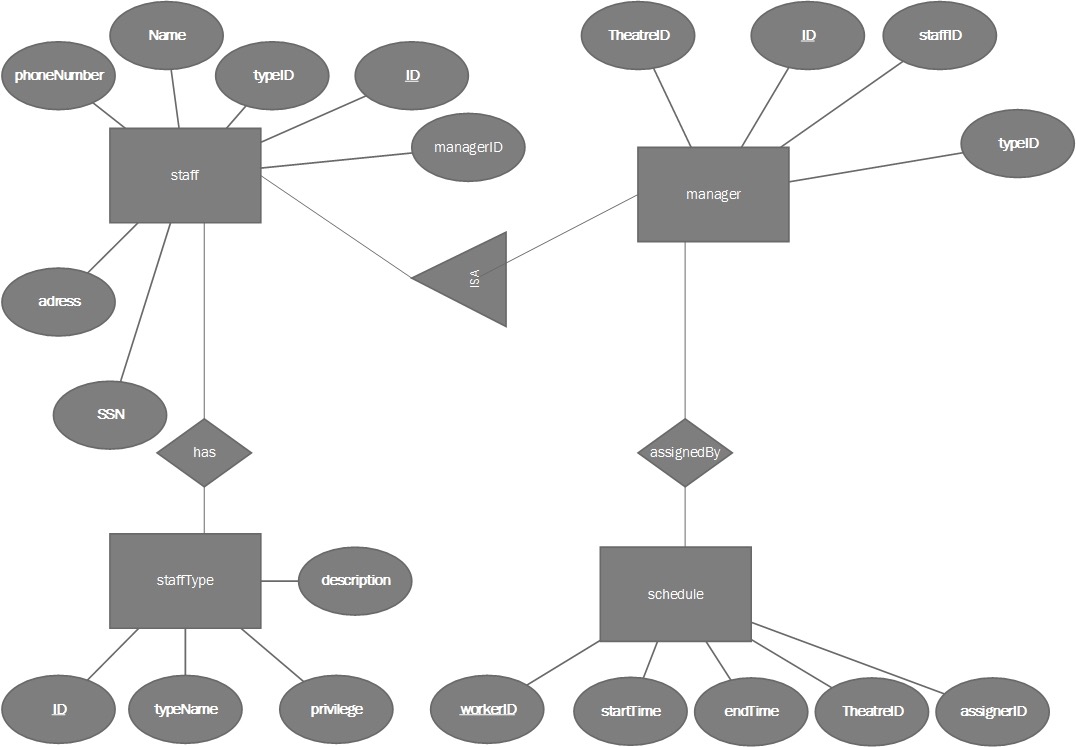
§ ER-Diagrams & SQL-Schemas

ER-Diagrams for project:

No.1 Theatre & Movie ER-Diagram

No.2 Member & Ticketing ER-Diagram

No.3 discussion Forum ER-Diagram

No.4 Staff Management ER-Diagram

/\*还有一些功能实现要说明 一般就是select xxx from xxx

我还没写这个。。。

\*/

SQL-Schemas:

**Following are the database schemas, all the primary keys are underlined.**

**Other constraints including candidate keys (CK), and foreign keys (FK) are specified.**

**Any attribute that is nullable is specified otherwise the attribute can not be null (not null).**

***Theatre*** (id, name, location)

**FK:**

***Movie*** (id, title, director, star, descrption)

**start nullable**

**ScreanRoom**(id, timeEnd, timeStart, TheatreID, MovieID)

**FK:** TheatreID references id in Theatre

MovieID references id in Movie

MovieType(id, typeName, MovieID)

FK: MovieID reference id in Movie

***Review***  (id, content, likes, authorUserID,)

FK: authorUserID refer to ID in member

***member*** (id, name, phoneNumber, emailID, creditCardNumber, regular)

regular nullable as a boolean

**regularMember**(id, userID, password, adress, creditCardExp, points, status)

points and status nullable

FK: id refer to id in member

***order***(id, memberID, screenRoomID)

FK: memberID refer to ID in member

screenRoomID refer to id in ScreenRoom

***Thread***(id, type, title, content, authorUserID, movieID, theatreID)

movieID, theatreID are nullable however type cannot be nullable.

type is a boolean , ‘This thread is a movie thread’

FK:

movieID refer to id in movie

theatreID refer to id in theatre

authorUserID refer to userID in regular member

***staff***(id, name, phoneNumber, ssn, addres, typeID, managerID)

FK:

typeID refer to id in staffType

mangerID refer to id in manager

暂时:both FKs nullable

***manager***(id, TheatreID, staffID, typeID);

Fk:

theatreID refer to id in theatre

staffID refer to id in staff

typeID refer to id in type

***staffType***(id, typeName, privilege, description)

***schedule***(workerID, startTime, endTime, theatreID, assignerID)

fk:

workerID refer to ID in staff

theatreID refer to ID in theatre

assignerID refer to id in manager