CS425 In Class Project document

Online Ticketing and Theatre Management System

Design & Implement

By

Bingyu Song

Jinyang Li

Xin Liu

§ 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
0.2	initial er-diagrams and schemas create.	Jinyang Li	10/11/2015
0.3	Correct and filled-up the schemas & Create feature explanation	Xin liu, Bingyu song	10/14/2015
0.4	Correct features explanation	Xin Liu	10/14/2015
1.0	filled up other contents	Jinyang Li, Bingyu song, Xin Liu,	11/2/2015
2.0	Final available system	Jinyang Li, Bingyu song, Xin Liu,	12/2/2015
extra	document editing own: Jinyang Li, Xin Liu, Bingyu Song /* part 1 is in the end of the doc. Also could get index at content table. */		

Feature Ownership

Feature	Tables In SQL	Owner
integer number is the feature id which you can find in end of document		
1-5	theatre, movie, screenRoom, movietype, member	Jinyang li
6-9	review, regularmember, oder, thread	Xin liu
10-13	stafftype, staff, manager, schedule	Bingyu Song

§Content Table

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

1. Introduction

1.1 Organization of Document

This document is oriented to explain our design process based the requirement of the Online Ticketing and Theatre Management System. There is a web application that is developed by PHP as the user interface. However, this document is concentrated on the designing of the database which is the key point to fulfill the system's requirement. And there are also a briefly statement about the web application.

In this document, the three important parts were including:

- ER-diagrams and SQL schemas
- Database design explanation
- Implementation

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

1.2 Requirements of project

The name of this project is:

<u>Online Ticketing and Theatre Management System</u>

This project require:

- Oracle database driver or any database driver which support Oracle database.
- unknown software system as host languages with 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 at least 3NF of database schema.
After analyzing the requirement, the 14 tables needed for this project. **<u>All tables are not violate BCNF.</u>**

Theatre

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

FD's: ID -> name,location

Movie

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	

FD's: ID -> title, director, star

Movie type

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	the description foe this kind of movie

FD's: ID -> type， description

Type_in

filed_name	data type	PK	nullable	description
ID	int	yes	no	unique identify number
typeID	int		no	foreign key refer to ID in movie_type
movieID	int		no	foreign key refer to ID in movie

FD’s: ID -> typeID， movieID

ScreenRoom

filed_name	data type	PK	nullable	description
ID	int	yes	not	unique identify number
timeEnd	time		not	end time of a movie
timeStart	time		not	starting time of a movie
TheatreID	int		not	foreign key refer to ID in Theatre table
MovieID	int		not	foreign key refer to ID in Movie table

FD’s: ID -> timeEnd, timeStart,. TheartreID, MovieID

Schedule

filed_name	data type	PK	nullable	description
workerID	int	yes	not	foreign key refer to ID in staff, unique identify number
startTime	time		not	start time of working
endTime	time		not	end time of working
theatreID	int		not	foreign key refer to ID in theatre
assignerID	int		not	foreign key refer to ID in manager

FD’s: workerID -> startTime, endTime, theatreID, assignerID

StaffType

filed_name	data type	PK	nullable	description
ID	int	yes	not	unique identify number
typeName	varchar(30)		not	the type of a job

privilege	boolean		not	whether this staff have privilege
description	varchar(100)		not	the descriptions of this type of job

FD's: ID -> typeName, privilege, description

Manager

filed_name	data type	PK	nullable	description
ID	int	yes	not	unique identify number
TheatreID	int		not	foreign key refer to ID in theatre
staffID	int		not	foreign key refer to ID in staff

FD's: ID -> TheatreID, staffID;

Staff

filed_name	data type	PK	nullable	description
ID	int	yes	not	unique identify number
name	varchar(30)		not	name of the staff
phoneNumber	int		not	phoneNumber of the staff
ssn	int		not	ssn of the staff
address	varchar(100)		not	address of the staff
typeID	int		yes	foreign key refer to ID in staffType
managerID	int		yes	foreign key refer to ID in manager

FD's: ID -> name, phoneNumber,ssn,address,typeID,managerID

Review table

filed-name	data-type	PK	nullable	decription
ID	int	yes	no	unique identify number
content	varchar(100)		no	movies saw before
likes	varchar(100)		no	favorite movies
authorUserID	int		no	foreign key refer to ID in member table
theatreID	int		yes	foreign key refer to ID in theater table

movieID	int		yes	foreign key refer to ID in movie table

FD's: ID -> content, likes, authorUserID, theatreID, movieID

Member table

filed-name	data-type	PK	nullable	decription
ID	int	yes	no	unique identify number
name	varchar(100)		no	name for one customer
phoneNumber	varchar(100)		no	phone-number for one customer
emailID	varchar(100)		no	email address for one customer
creditCardNumber	int		no	credit card number for one customer
regular	boolean		no	whether this customer be regular or not

FD's:ID -> name,phoneNumber, emailID, creditCardNumber, regular

regularMember

filed-name	data-type	PK	nullable	decription
ID	int	yes	no	foreign key refer to ID in member table
user_ID	int	yes	no	unique identify number
password	int		no	unique password for customer
address	varchar(100)		no	address for this customer
creditCardExp	data		no	express date for the credit card
points	int		yes	
status	varchar(100)		yes	

FD: ID, user_ID->password, address, creditCardExp, points, status

order

filed-name	data-type	PK	nullable	decription
ID	int	yes	no	unique identify number
memberID	int		no	foreign key refer to ID in member table
screenRoomID	int		no	foreign key refer to ID in ScreenRoom table

FD: ID->memberID, screenRoomID

Thread

filed-name	data-type	PK	nullable	decription
ID	int	yes	no	
parentThreadID	int		yes	
title	varchar(100)		no	
content	varchar(100)		no	
authorUserID	int		no	foreign key refer to userID in regular member
movieID	int		yes	foreign key refer to ID in movie table
theatreID	int		yes	foreign key refer to ID in theater table

FD: ID-> parentThreadID, title, content, authorUserID, movieID, theatreID

3 Implement

3.1 Physical structure

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

3.2 Safety

After analyzing the requirements, there're 2 database accounts needed.

First is owner which has DBA privileges, which could be done like this:

SQL> Create user owner

identified by ownerpass

SQL> grant connect, resource,dba to owner

now the owner account has right to delegate rights to other accounts and all the physical structure implement would be done under owner account.

After create owner, the website Administrator account should be created and using in web host languages.

SQL> Create user webadmin

identified by adminweb

SQL> grant connect, resource to webadmin

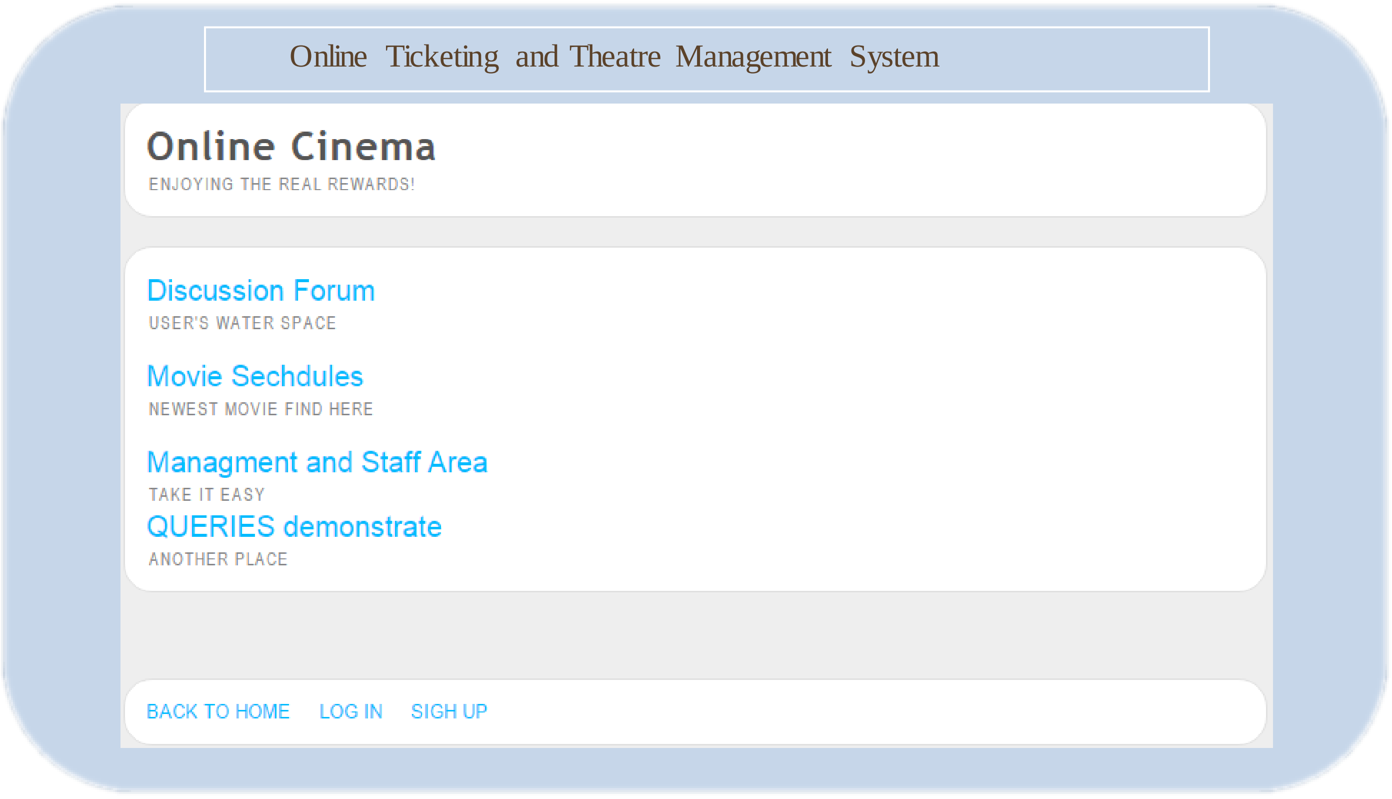
3.3 Web application and the Database

Using PHP to develop a web application.

The tiggers and database schema SQL script files (*.SQL) needs to be manually run in SQL developer in orders:
create.SQL -> insert.SQL -> triggers.SQL

Then combine the SQL query statement with the php.

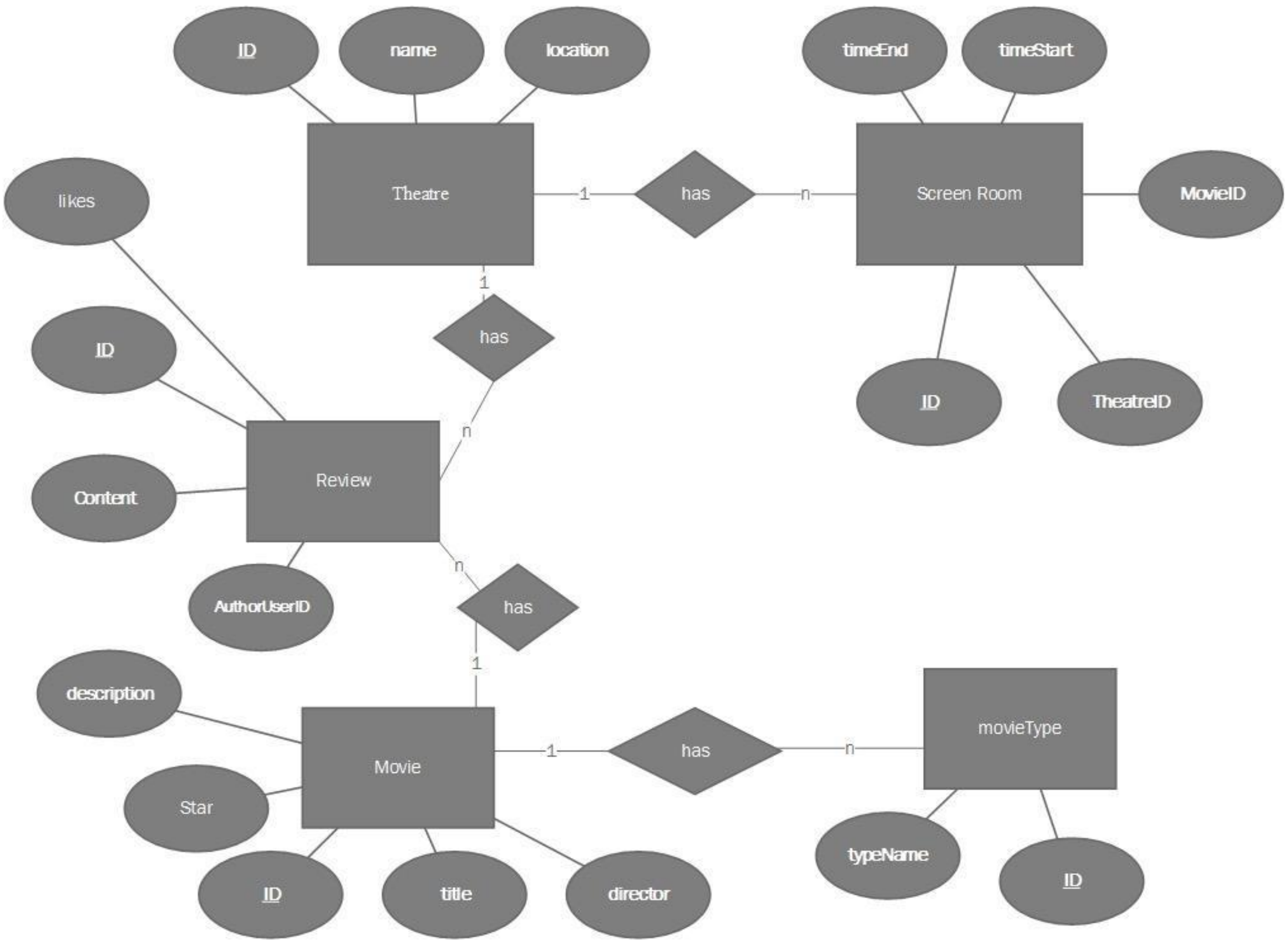
Following is the outlook of the web application:



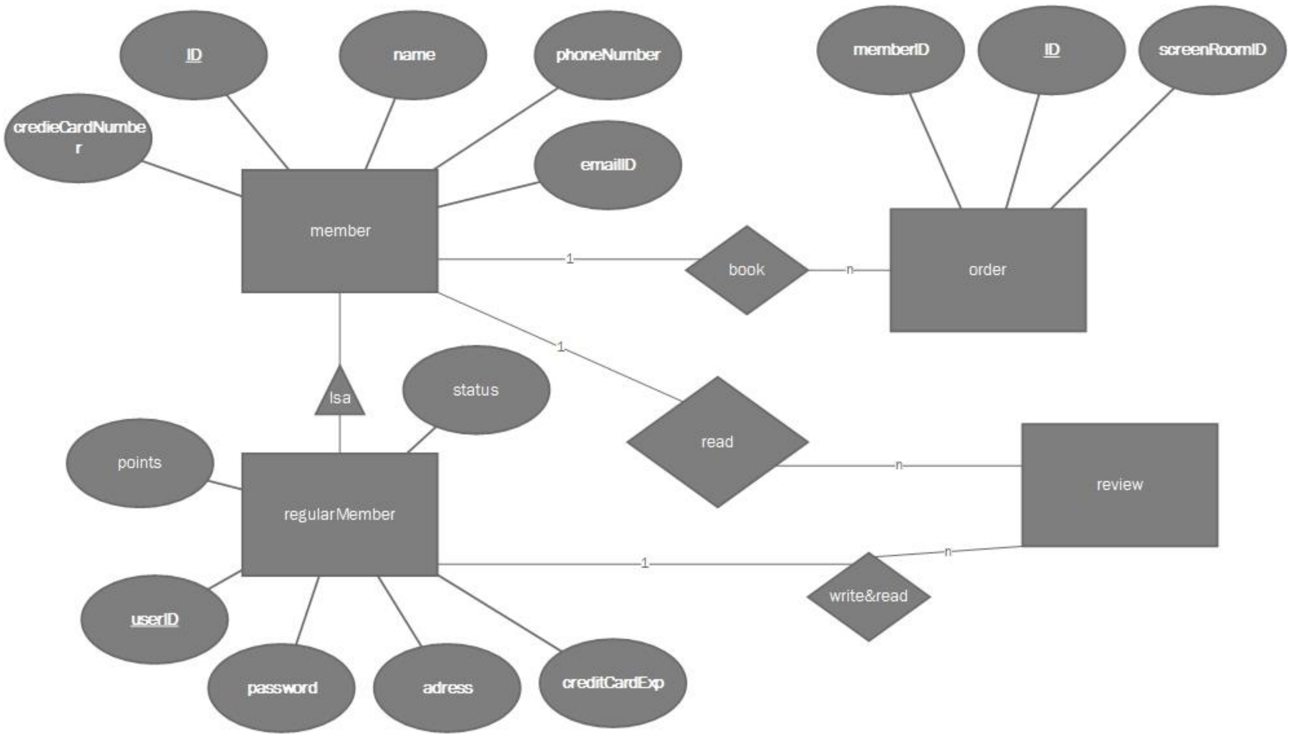
§ ER-Diagrams & SQL-Schemas

ER-Diagrams for project:

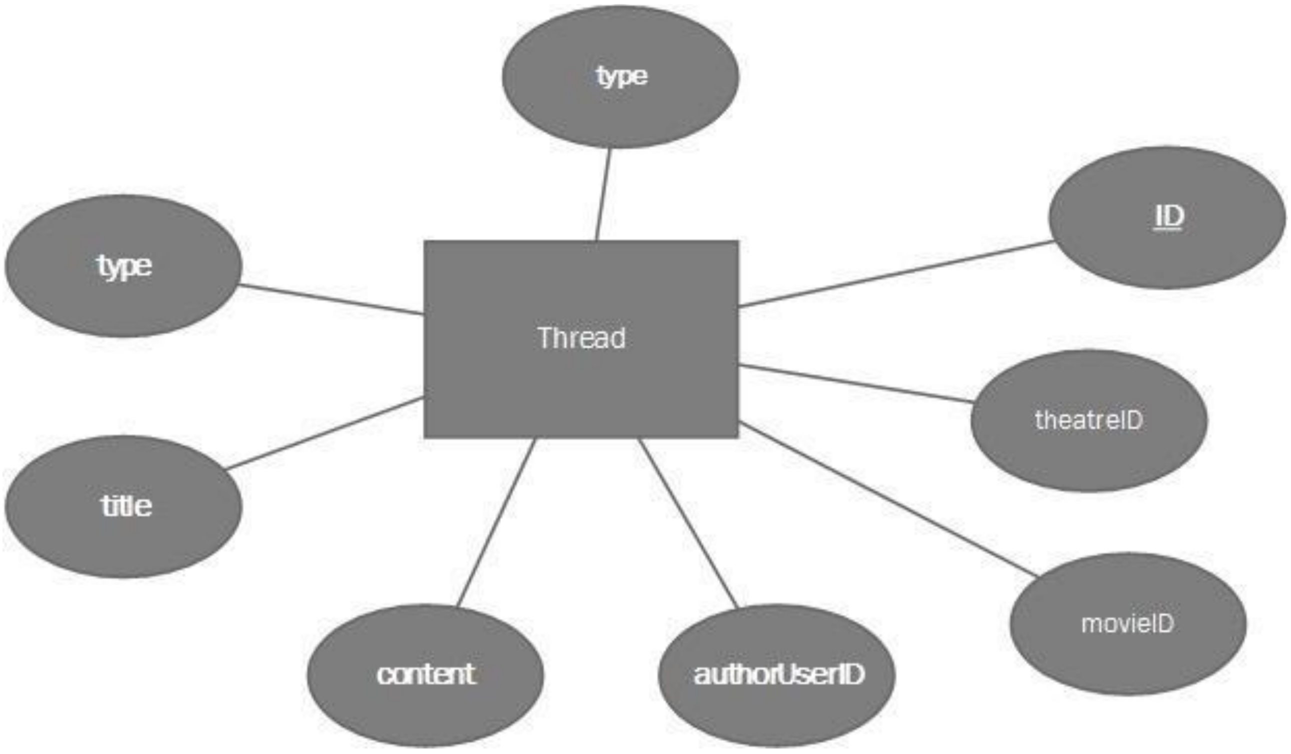
No.1 Theatre & Movie ER-Diagram



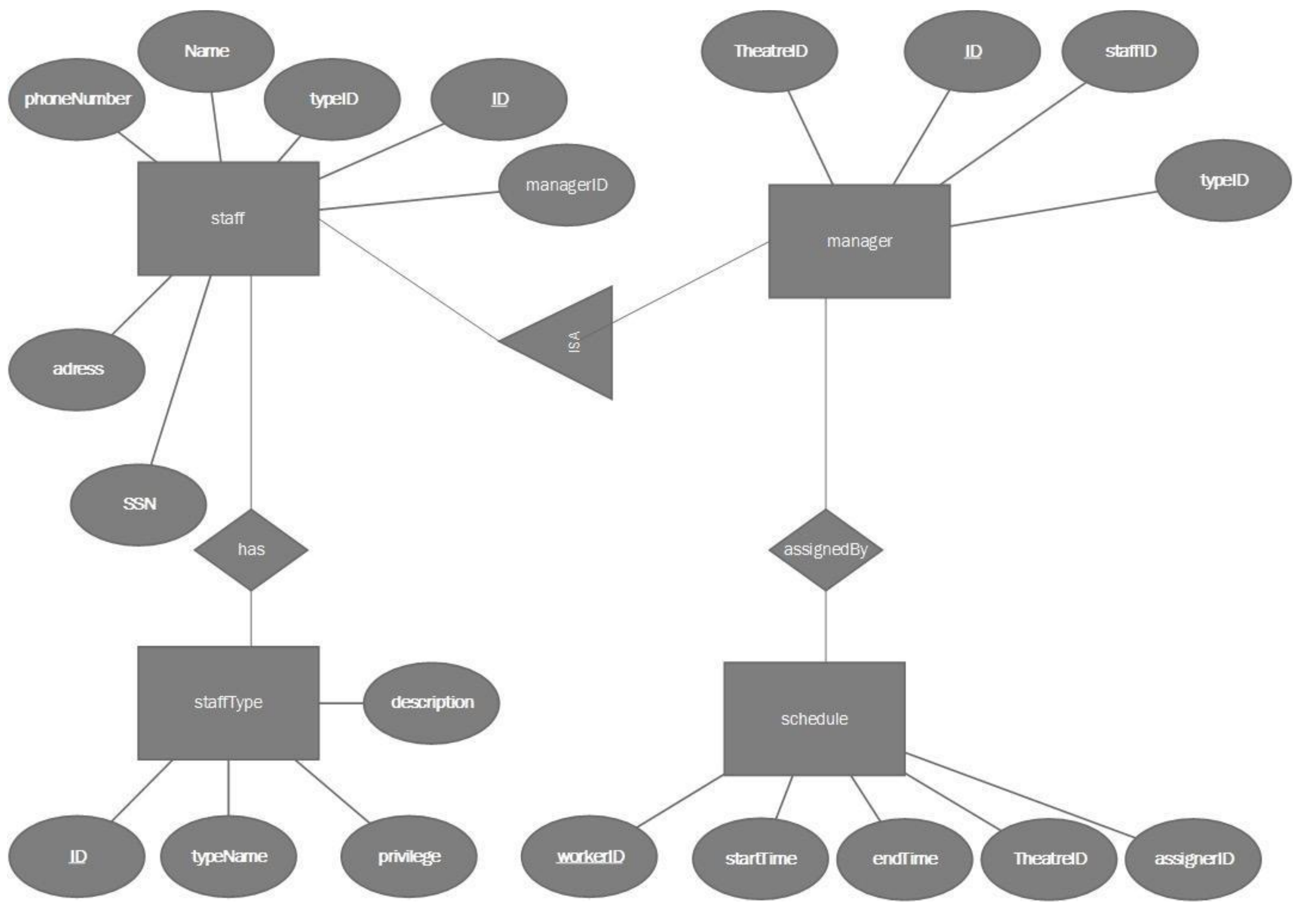
No.2 Member & Ticketing ER-Diagram



No.3 discussion Forum ER-Diagram



No.4 Staff Management ER-Diagram



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 (<u>id</u>, name, location)

Movie (<u>id</u>, title, director, star, descrption)
star nullable

ScreenRoom (<u>id</u>, timeEnd, timeStart, TheatreID, MovieID)
FK: TheatreID references id in Theatre
MovieID references id in Movie

MovieType (<u>id</u>, typeName, descrption)

Type_in(<u>id</u>, typeID, movieID)
FK:
typeID refer to id in MovieType
movieID refer to id Movie

Review (id, content, likes, authorUserID,theatreID movieID)
FK:
authorUserID refer to ID in member
movieID refer to id in movie
theatreID refer to id in theatre
both movieID and theatreID nullable

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

orders(id, memberID, screenRoomID)
FK: memberID refer to ID in member
screenRoomID refer to id in ScreenRoom

Thread(id, parentThreadID, title, content, authorUserID, movieID, theatreID)
movieID, theatreID are nullable however type cannot be nullable.
parentThreadID is an int , which could be null, point to the parent thread to implement follow-up function.

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, address, typeID, managerID)
FK:
typeID refer to id in staffType
mangerID refer to id in manager
both FKs nullable

manager(id, TheatreID, staffID);
FK:
theatreID refer to id in theatre
staffID refer to id in staff

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

feature support：

1.use 'Theatres' table to store a theatres address, name etc.

2.use 'movies' table to store movies' titles,stars, directors, description etc.

3.since theatres may have several screens, bring up 'screen room' table.
  use fk 'theatre id' to refer the pk--'id' of 'Theatres' table to define which theatre does this screen room in.
  use fk'movie id' to refer th pk --'id' of 'Movies'table to define which movies this screen is playing.
  use attribute ‘timestart’,’timeend’ to define the movies’ specific schedule.

4.since a movie can be of several types, bring up 'movietype' table.
  one type also can belong to several types . use table ‘type_in’ to define this connection.

5.use 'member' table store guest or vistor's information.there name,phone number etc.
attribute ‘regular’ is a boolean which define whether this guest have registerd or not.

6.use 'Review' table store reviews on the theatres and the movies.
use 'authoruserid' to refer the pk--'id' of 'member' table to define who writes the review.
use another two fk to implement the feature ‘a review can under theatre and movie’, in application level, can easily using some queries to get the movie information and the information of theatre.

7.use 'regularmember' table to store information of a regested guest like user id, password, credit point etc.
this table share same pk with 'member'table, and is a subclass of 'member'.

8.since guest can book a tickets, use 'order' table to store this kind of information.
  use fk 'member id' to refer the primary key of 'member'table to define who bought this tickets.
  use fk 'screenroomid' to refer pk in 'Screenroom' table to define the detailed information of this ticket.
  use the information in 'order'table ,we can gather the movies name being screened by a guest. Futhermore, we can caculate the type of movies in the last 30days.

9.use 'thread' table to store the discussion topics.
  use a boolean attribute--'type' to define this thread is a movies group or a theatre experience group.
  use two fk to refer to the pk in 'Theatre' and 'Movies' table to define its characteristic.
  use fk to refer the pk in 'membership' table to support the function that a registered user may get credit points based on his activites.

10.use 'stafftype' table to store descriptions of a kind of staff which including owner, managers, ticketing staff and so on.

'the owner and the web administrator have all the privilege to see and update all the tables and the data in the tables. The owner can delegate privileges to employees in the theatre'-------- this function is supported by the attritubte 'privilege'.

11.use 'staff' table to store staff members' detail information like his name, address, phone number, location where he works and ssn.

use fk 'typeid' to refer to pk in 'stafftype'table to define which kind of job does this staff do.

use fk 'managerid' to refer to pk in 'manager'

12.use'manager' table to store information about manager,owner and web administrator.

use fk staffID to refer to id in 'staff' to define get the type of this job.

13.use'schedule' table to store information about schedule of the emplyees.

to support the function--'the schedule of the employees will be decided by the managers or the owner'.

use fk 'assignerid' to refer to pk in 'manager' table.

14. to implement the management right, it would be done using database account privilege and should be associated with host languages.