CS2102 Progress Report
Online Booking System

Movie Tickets Booking System

Hu Wenyan (A0119397R)

Lu Yuehan (A0119387U)

Luan Wenhao (A0119541J)

20 September 2014

Overview

In this project, we aim to design a online movie tickets booking application to facilitate convienient searching and booking for users. User will be allowed to search catalogue of movie based on the **movie title**,**movies attributes(such as actors, director and movie descriptions)**,**time slots**,**cinema name**,**cinema location**, and **ticket price range**. In addition,after logging in, users are able to book movie tickets, and modify or cancel their booking afterwards.

Implementation

ER Diragram

- **Description** In this ER diagram, there are five entities including **User**, **Time Slots**, **Movie**, **Cinema** and **Hall**.
  - **User**

    Users are distinguished by their user ids which are allocated by system during account registration. User types includes **normal**,**senior** and **student**. Certain types of users are able to enjoy discount price for movie tickets.
  - **Time Slots**

    For each movie, there are several time slots which contain movie title, cinema, hall in cinema,number of available tickets and standard ticket price.
  - **Movies**

    Movies have seven attributes including year, title, actors, director and descriptions; movies are distinguished by its title and year.
  - **Cinema**

    Cinemas are distinguished by its name and location.
  - **Hall**

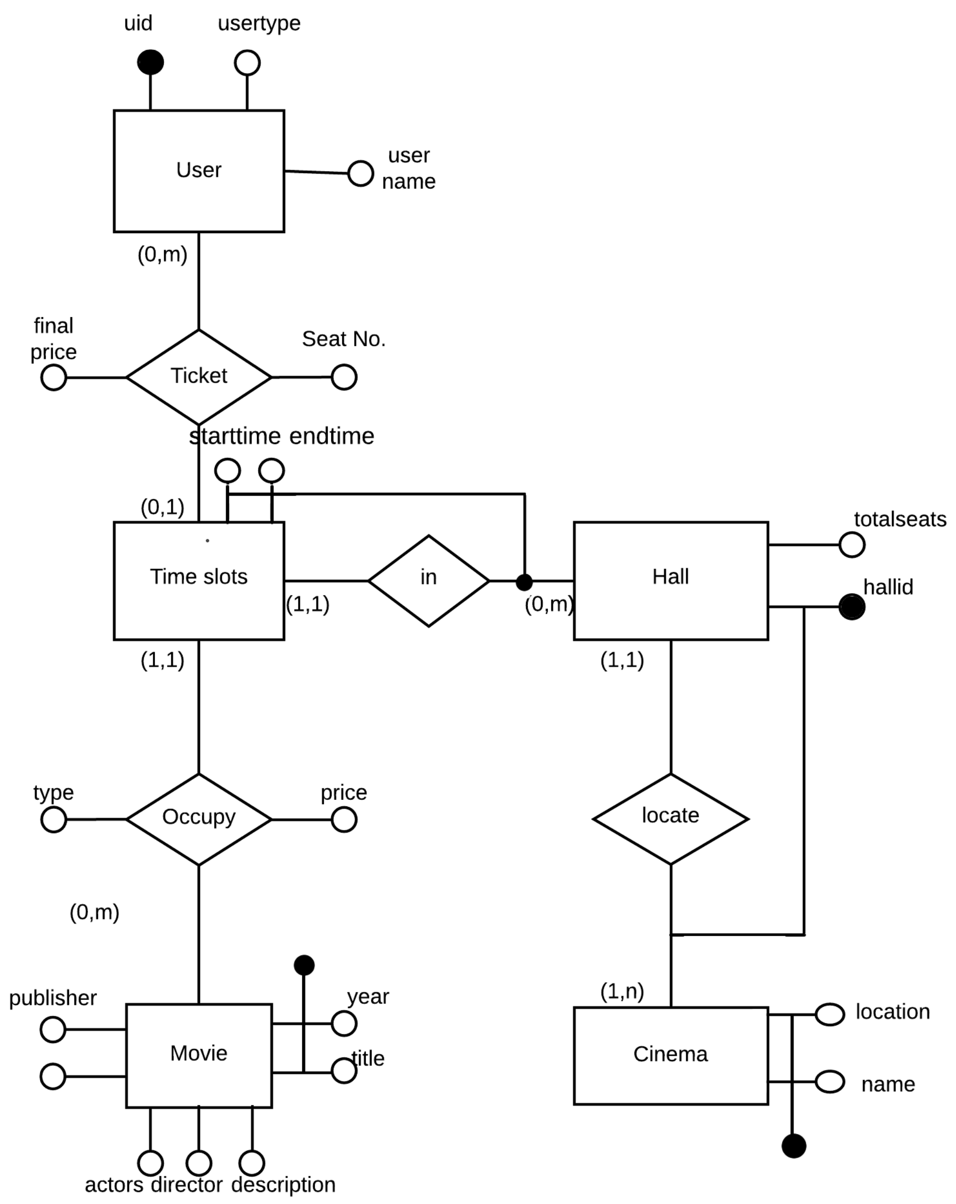
    Hall is a weak entity and depends on cinema.
  - **Ticket**

    Users can book tickets within a time slot.The final price of a ticket will take user's type into consideration. It is a many to many relationship, that is to say, one user can book more than one tickets within same time slot, and for each timeslot, there might be zero to many tickets being sold.

- **Occupy**

  For every timeslot, it must be occupied by a movie, but for movie, it may have no slot or many time slots.

- **Diagram**



Relational Schema

- SQL DDL Code
  - Cinema

```
CREATE TABLE cinema (
name VARCHAR(128),
location VARCHAR(128),
PRIMARY KEY(name, location)
);
```

  - Hall

```
CREATE TABLE hall (
hallid INTEGER PRIMARY KEY,
totalseats INTEGER DEFAULT 100,
nameofcinema VARCHAR(128) NOT NULL ,
locationofcinema VARCHAR(128) NOT NULL,
CONSTRAINT fk_cinema
FOREIGN KEY(nameofcinema, locationofcinema) REFERENCES
cinema(name, location) ON DELETE CASCADE
);
```

  - Timeslot

```
CREATE TABLE timeslot(
starttime DATE,
endtime DATE,
hallid INTEGER,
PRIMARY KEY(hallid,starttime,endtime),
CONSTRAINT fk_hall
FOREIGN KEY(hallid) REFERENCES hall(hallid) ON DELETE CASCADE
);
```

  - Occupy

```
CREATE TABLE occupy(
price NUMERIC,
movietype VARCHAR(64),
movietitle VARCHAR(128),
movieyear INTEGER,
starttime DATE,
endtime  DATE,
hallid INTEGER,
PRIMARY KEY(movietitle, movieyear, starttime, endtime, hallid),
CONSTRAINT fk_movie
FOREIGN KEY(movietitle, movieyear) REFERENCES movie(title,year),
CONSTRAINT fk_timeslot
FOREIGN KEY(hallid,starttime,endtime)
REFERENCES timeslot(hallid,starttime, endtime)
ON DELETE CASCADE
);
```

- Movie

```
CREATE TABLE movie(
title VARCHAR(128),
year INTEGER,
actors VARCHAR(256),
director VARCHAR(64),
description VARCHAR(512),
producer VARCHAR(128),
PRIMARY KEY (title, year)
);
```

- User

```
CREATE TABLE subscriber(
subscriberid CHAR(16) PRIMARY KEY,
username VARCHAR(32),
usertype VARCHAR(32),
CONSTRAINT fk_subscriber
CHECK (usertype = 'STUDENT' OR usertype ='NORMAL'
OR usertype = 'VIP' OR usertype= 'SENIOR')
);
```

- Ticket

```
CREATE TABLE ticket(
subscriberid CHAR(16) REFERENCES subscriber(subscriberid),
starttime DATE,
endtime  DATE,
hallid INTEGER,
PRIMARY KEY(subscriberid, starttime, endtime, hallid),
CONSTRAINT fk_timeslots
FOREIGN KEY(starttime, endtime, hallid) REFERENCE
timeslot(starttime,endtime, hallid)
);
```