CS425 - Fall 2013 - Course Project
A Cinema Management System

Boris Glavic

October 6, 2014

1 Project Timeline

The project has three deliverables. Deadlines are announced on the course webpage `http://cs.iit.edu/~cs425/schedule.html`. This document can be downloaded from `http://cs.iit.edu/~cs425/files/projectdesc.pdf`. Each group will **demo** their application at the end of the semester. The deliverables are:

- **ER-model**: Each group should develop an ER-model for the application. This can be uploaded as any type of image file (please do not use esoteric formats).
- **Relational schema**: The second deliverable is a translation of the ER-model into a relational schema implemented as an SQL script. The script should use Oracle's SQL dialect and should execute without errors on the `fourier.cs.iit.edu` Oracle accounts. Besides from defining tables and constraints, this script should create indexes where appropriate. Please upload the script as a simple text file.
- **Application**: The last deliverable is a cinema management application that uses the relational schema defined in the first two deliverables. This application can be either a web or desktop application.

Some of the requirements are marked as optional **bonus** requirements. You are free to not realize these requirements, but you can get extra credits by implementing them.

Every member of the group has to contribute in each phase of the project and you will be graded on your individual contribution and on the overall project result.

2 Overview

The goal is to build a cinema management application using a database backend to store information about movies, rooms, tickets, and so on. The application will support two types of users. Clients of the cinema can search for movies and show times, book tickets, and rate movies. Staff of the cinema can modify movies, showtimes, and room information, and query user information. For the project we record information for a **single cinema** and its users.

3 Data Requirements

3.1 Users

We will consider two types of users:

- **Members**: Members are the clients of the cinema. We should record the name of each member.
- **Staff**: Staff of the cinema.

3.2 BONUS: Payment Methods

Members can have one of more payment methods registered (for buying tickets), e.g., a credit card.

3.3 Movie Information

The database should record information about movies, actors, directors, and so on.

- **Movies**: A movie has a title, one or more genres, a release year, a length (in minutes), and a PG rating.
- **Actors**: Actors play roles in movies. Note that actors may play more than one role in the same movie (e.g., *Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb*).
- **Director**: Directors direct movies. A movie may have more than one director.
- **Writer**: Writers write movies (scripts).
- **Person**: Actors, Directors, and Writers are all persons. A person has a name, a data of birth, and a gender.

3.4 Cinema Information and Ticketing

The cinema has several show rooms and each of these rooms shows zero or more movies each night. Members can check availability of tickets for certain showings and book tickets.

- **Room**: A room has a number and a capacity (maximum number of persons that can watch a movie at the same time).
- **Showing**: Movies can are shown in a certain room at a certain time. There cannot be more than one show in each room at a given point in time. There may be multiple showings for the same movie at the same time (albeit in different rooms).
- **Tickets**: A ticket belongs to a member (who brought the ticket) and is valid for a specific show time. A ticket also has a price.

3.5 BONUS: Rating System

Store ratings for movies and actors. Individual ratings should be associated with a member.

4 Application Requirements

The application should support the following actions. We indicate for each action whether it can be executed by staff or members.

- Search for movie, actor, director, writer information (`staff/member`)
- Check availability of shows for a movie based on time (`staff/member`)
- Buy tickets (`member`)
- Add/Delete/Modify movie, actor, director, writer information (`staff`)

4.1 Search

The central component of the application is the movie search interface. The application should support different search criteria (e.g., by director, by title, by genre, by actor) and combinations of these search strategies. *For example, return all movies that are written by Stanely Kubrick which are of genre horror.* Both complete matches and partial matches should be supported. The results of a search should be shown as a list of movies to the user. The user should then be able to access detailed information for each movie or refine/modify his search. A movie information page should show the information we store for the movie and related information such as director(s), wirter(s), and actor(s) (for each actor the role(s) he/she plays).

4.1.1 BONUS: Saved Searches

Enables the user to store searches and re-execute them later.

4.1.2 BONUS: Movie information page with images

Also show posters for movies and photos for actors and directors.

4.1.3 BONUS: Display summary information with persons and person centric information pages

In a movie information page you should also show for each involved person some summary information such as number of movies directed/played and so on. Also provide information pages for each person that lists the movies they have participated in and what was their responsibility in each movie (director, actor, writer).

4.2 Buy Tickets

Members can buy tickets for a certain showing of a movie. This should be accessible through the movie information page.

4.3 Add/Delete/Modify Movie Information

Staff of the cinema should be able to add, delete, and modify movie information (and persons).

4.3.1 BONUS: modify room and showtimes information

Allow staff of the cinema to modify room and showtimes information, e.g., add new showtimes for a movie.

4.3.2 BONUS: IMDB lookup

Allow staff of the cinema to search for movies on IMDB and import this information into your database.