Logan Saso - Alex Grasley

http://web.engr.oregonstate.edu/~grasleal/index.html

Project Step 3 - Modified Feedback

A - Feedback Adjustments

New Changes:
Completely reworked the design of the webpage. Moved from multiple pages to tabs on a single page. Added search features for all tables based on feedback. Also changed form fields to include drop down menus when selecting items from other tables that would be difficult/impossible to enter the correct foreign keys. Tables in the code are only one template row for use with Jinja and Flask. Updated modal forms to be more consistent and used a lighter colored theme all based on peer feedback.

Previous Changes:
Made some adjustments to the Schema to reflect key links (with underlines).
Discussed with TA about many to many relationship (through tickets). Consider tickets a linker table with the additional property that we include a price.

Some feedback received regarding adding all attributes to the ER diagram directly conflicted with the assignment description, so no changes were made.

B - Project and Database Outline

Outline

We're designing a replacement for movie-ticket tracking/sales websites like Fandango. We believe, here at GoodMovies INC, that we can provide the best movie-going experience and maximize ticket sales for participating theaters. Our website will show movies playing at various theaters, tracked by room and tracking ticket sales.

Database Outline

Entities: (if no constraint listed, the attribute has no constraints)

1. Movies
    a. Attributes
        i. MovieID - PRIMARY KEY AUTO-INCREMENTING INT
        ii. Name - VARCHAR(255) NOT NULL

Logan Saso - Alex Grasley

http://web.engr.oregonstate.edu/~grasleal/index.html

   iii. Rating - ENUM (E, PG, PG-13, R, AO)
   iv. Genre - VARCHAR(255)
   v. Length - UNSIGNED INT(16) NOT NULL
b. Relationships
   i. A movie can have many showings

2. Theatres
   a. Attributes
      i. TheatreID - PRIMARY KEY AUTO-INCREMENTING INT
      ii. Name - VARCHAR (255)
      iii. Address - VARCHAR(255)
         1. Address through Zip must exist and be consistent with each other. IE, if an address doesn't exist in the right zip code it shouldn't be added to the table.
      iv. Address 2 - VARCHAR(255)
      v. City - VARCHAR(255)
      vi. State - VARCHAR(255)
      vii. Zip - VARCHAR(32)
   b. Relationships
      i. A theater has one or many rooms
   c. Constraints
      i. A theatre must have at least one room
3. Rooms
   a. Attributes
      i. RoomID - PRIMARY KEY AUTO-INCREMENTING INT
      ii. Theatre - FOREIGN KEY NOT NULL
         1. A room is related to exactly one theatre
      iii. Capacity - UNSIGNED INT(8) NOT NULL
         1. Capacity must be > 0
   b. Relationships
      i. Rooms have one movie
      ii. Rooms can have multiple showings
4. Showings
   a. Attributes
      i. ShowingID - PRIMARY KEY AUTO-INCREMENTING INT
      ii. RoomID - FOREIGN KEY
         1. Showing MUST have one and only one room
      iii. Time - DATETIME (NOT NULL)
         1. Showing must have a date and Time
         2. Showtimes must not overlap in the same room (taken care of by the scheduler)
      iv. MovieID - FOREIGN KEY (NOT NULL)

Logan Saso - Alex Grasley

http://web.engr.oregonstate.edu/~grasleal/index.html

1. Showings must have one and only one movie

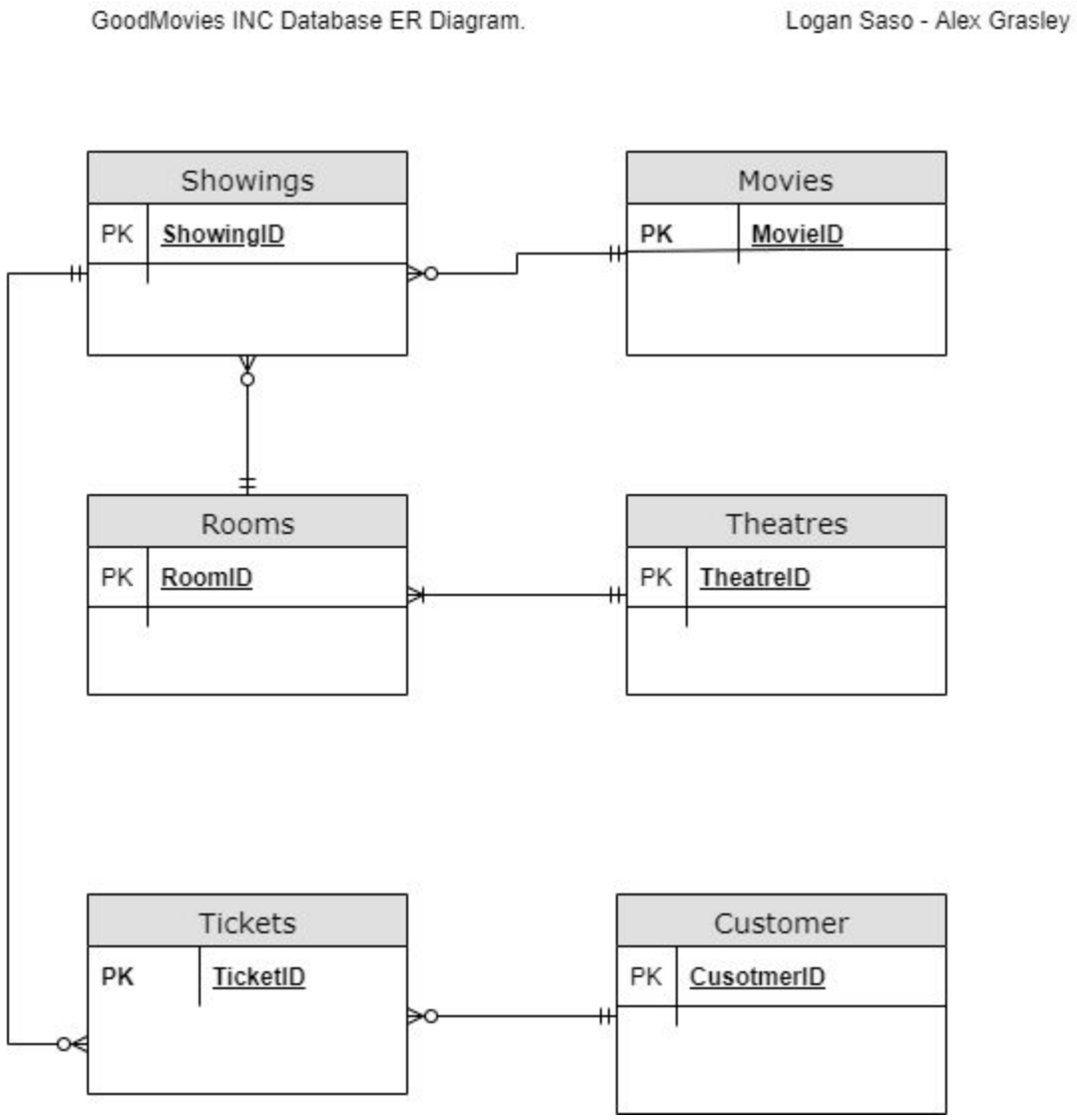
b. Relationships
   i. A showing has many customers

5. Tickets
   a. Attributes
      i. PRIMARY KEY combination of ShowingID and CustomerID
      ii. Price - DECIMAL (NOT NULL)
      iii. ShowingID - FORGEIGN KEY NOT NULL
         1. Ticket must have one and only one ShowingID
      iv. CustomerID - FORGEIN KEY
         1. Ticket can only have one Customer
   b. Relationships
      i. One ticket has one showing
      ii. One ticket has one customer
   c. Additional Note:
      i. Tickets acts as the linker table between Customer and Showing to facilitate the Many to Many relationship between them. We realized that we could also use this table to store more useful information, such as “price” in this case.
6. Customer
   a. Attributes
      i. CustomerID - PRIMARY KEY AUTO-INCREMENTING INT
      ii. LName - VARCHAR(255)
      iii. FName - VARCHAR(255)
      iv. BirthDate (DATE)
   b. Relationships
      i. A customer can go to many showings
      ii. A customer can buy many tickets

Logan Saso - Alex Grasley

http://web.engr.oregonstate.edu/~grasleal/index.html

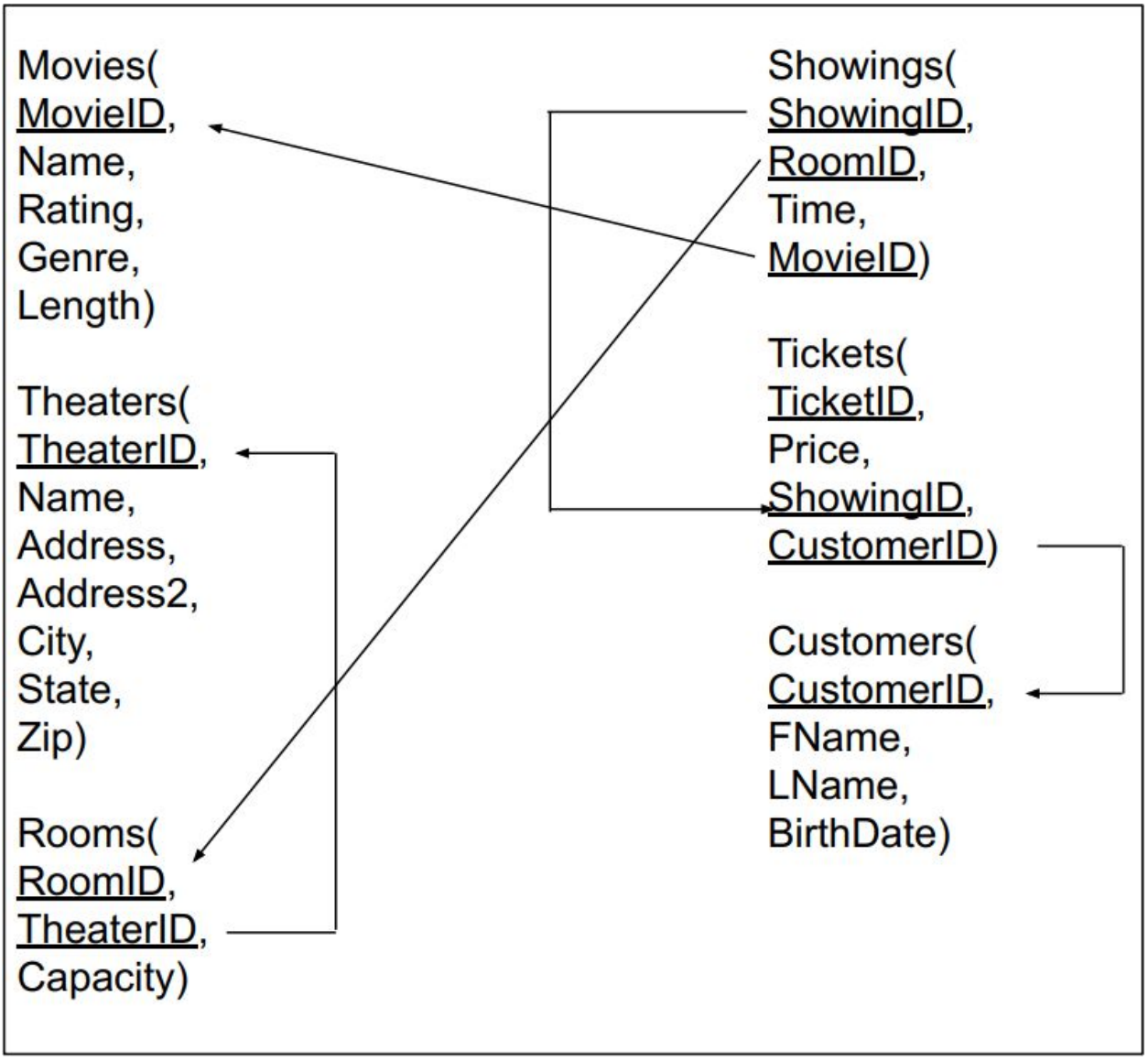
C - ER Diagram



Logan Saso - Alex Grasley

http://web.engr.oregonstate.edu/~grasleal/index.html

D - Schema



Note: In our many to many relationship we realized there were some properties aside from primary keys that would be useful to store for every connection. Because of this we use the Ticket table to store the M-M relationship between Customers and Showings.