Kaylee Moore and Jack Lindner  
**Game Rental Database**  
CS 461, Database Systems Final Project  
Professor Sutton

This project was designed to help students learn how to create a database management system from scratch. Given a project PDF, we set out to create an in-depth database setup to demonstrate our skills in database management systems (DBMS).

Our goal with this project was to provide a system that manages the inventory and rental of video games in a rental store network of franchises. In this franchise system, there are many stores, each in a different region carrying different stock and having different employees and customers.

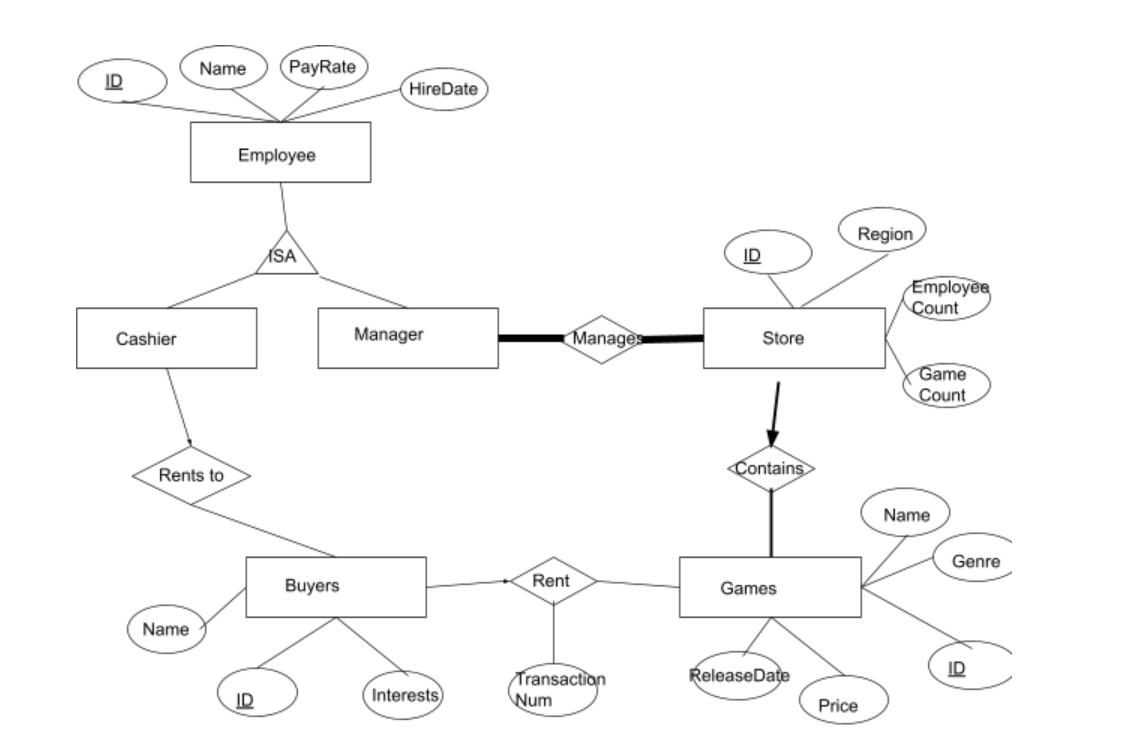
One of the main technical problems we faced when designing this database system was regarding the staff. To differentiate managers from cashiers, we needed to make specific logins for managers that have administrative permissions, but only for the managers.

Some of the main requirements of this database were: buyers register with their name and interests, buyers can view game information, buyers can search for games at either their region’s store or different store, cashiers and managers can check out buyers, buyer can see cashiers name but no other information, everyone can see if game is rented out currently, cashiers and managers can view all store information, buyer can only view store region and game count, managers can edit inventory manually, and cashiers can but all game information but not edit the game information.

In order to mitigate redundancies, we decided to break up all the items as we built the schema. We did this by choosing not to make a single entry for each game and have a bunch of attributes, making everything self-contained instead. This eliminates overlap and redundancies between the tables which leaves less room for errors and unexpected outputs.

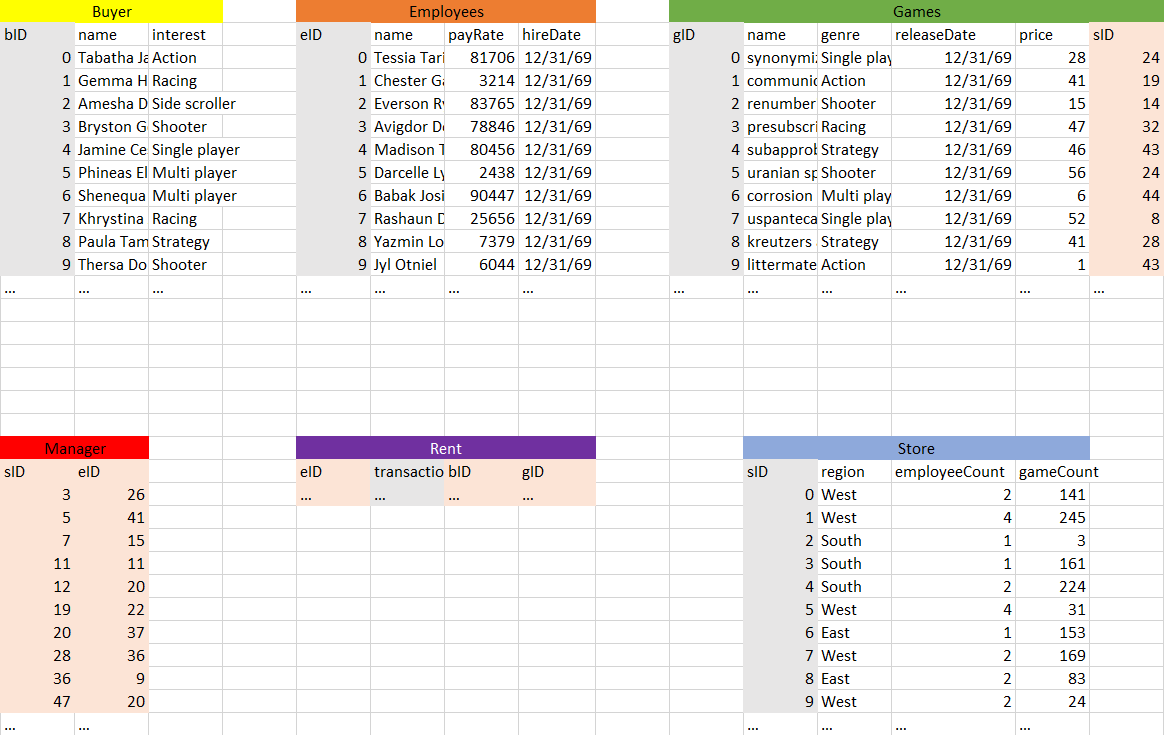
When designing the GUI, we went for a very simple and easy-to-follow design. With five buttons along the top and a dropdown selection, you can specify which table you would like to view and then act upon that data. Some actions available currently are add, delete, and modify entries, and move between pages using “previous” or “next” buttons.

ER Diagram:



We decided to break up the Entity-Relationship diagram in the fashion above so that we would reduce functional dependencies and possible weak points in our database.

BCNF of tables:



To generate the random data in the database, we used two sets of Internet-provided lists and one list generated by Jack Lindner. The first list, genres.txt, was generated by Jack and has a bunch of random game genres that he could think of off the top of his head. The two others, names.txt and words\_alpha.txt, are both pulled of the Internet. We then used those three lists and the Java random number generator to randomly pick lines from each of these files to generate different things like renter names, game names, and game genres. To generate the pay rate of employees, we used the Math.random function to generate a number from 0 to 100,000. To generate the date for employee hiring and game release dates, we generated a random integer for each part of the date: 1 through 12 for month, 1 through 28 for day, and 1 through 2019 for the year and then assembled this into a string.

The user-interface for this database is laid out in a straightforward way. Along the top bar, there are five buttons and a table dropdown. Below these buttons is the table showing all the information in the table selected using the table dropdown. The **New** button will show three dropdowns, one for “Employee ID”, “Game ID”, and “Buyer ID” respectively. Clicking the save button to the right of the “Buyer ID” button will save your entry to the database, creating all the appropriate relationships. The **Delete** button will delete the top-most row in the selected table using the table dropdown. The **Update** button will push any updates to the database that you have made by selecting and editing an entry. The **Next** button will move to the next table. The **Previous** button will move to the previous table.

Project source code:

DatabaseUI.java

GenerateRandom.java

Discussion: