University Housing Database

CIS 451 Final Project

Syd Lynch

Contents

Links

Home Page: http://ix.cs.uoregon.edu/~slynch2/CampusHousingDB/home.php

(All code is linked at the bottom of the home page. Each PHP file is linked as a text file)

Administrative Tools –

Roster: http://ix.cs.uoregon.edu/~slynch2/CampusHousingDB/roster.php

Look-Up: http://ix.cs.uoregon.edu/~slynch2/CampusHousingDB/lookup.php

Check-In Check-Out: http://ix.cs.uoregon.edu/~slynch2/CampusHousingDB/cico.php

Equipment Check-out –

Loan: http://ix.cs.uoregon.edu/~slynch2/CampusHousingDB/equipLoan.php

Return: http://ix.cs.uoregon.edu/~slynch2/CampusHousingDB/equipReturn.php

Key Check-out –

Loan: http://ix.cs.uoregon.edu/~slynch2/CampusHousingDB/keyLoan.php

Return: http://ix.cs.uoregon.edu/~slynch2/CampusHousingDB/keyReturn.php

Packages –

Log: http://ix.cs.uoregon.edu/~slynch2/CampusHousingDB/packageLog.php

Deliver: http://ix.cs.uoregon.edu/~slynch2/CampusHousingDB/packageDeliver.php

Summary

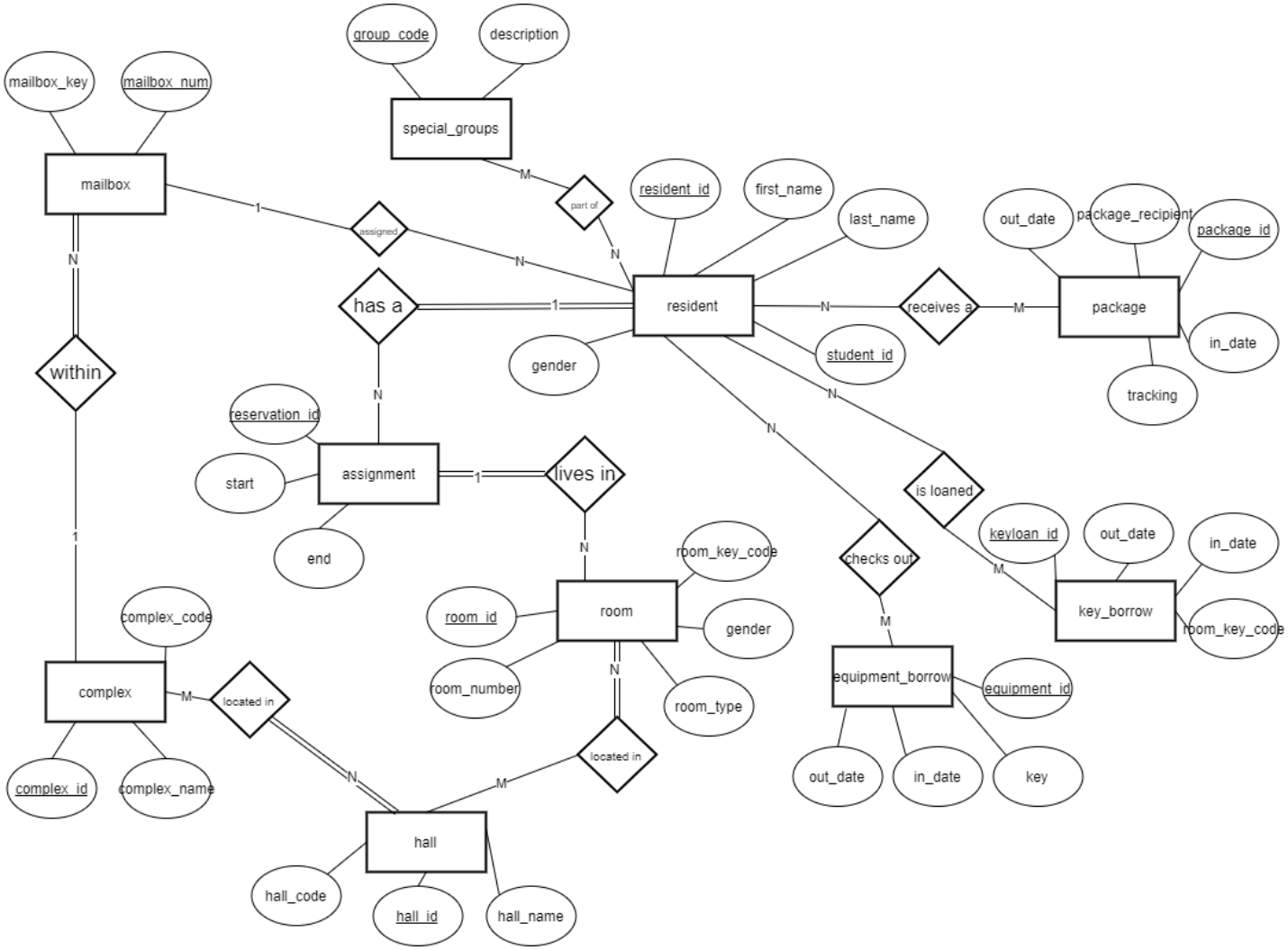
My database will maintain information for a fictional university's campus housing system. This information would be used primarily by Housing staff as a method of interacting with and overseeing residents that live on campus. This includes a look-up function that will display a resident's housing assignment, a room roster, and modules for checking in or checking out a resident, equipment loans, key loans, package logging, and package delivery.

The database will be largely driven by the `resident` table which will consist of a unique integer identifier, a student ID number, first and last name, gender, and any special groups they are associated with. Every resident on campus will have an entry in the `assignment` table. On the university's campus there will be several unique complexes, each containing multiple halls, and within each hall are many rooms. The `assignment` table will map each resident to a complex, hall, and room, containing other various information such as their check-in date, and check-out date. Other tables serve as methods of logging interactions with residents: `equipment_borrow` maintains a record of any equipment loaned to a resident, `key_borrow` contains records of any keys loaned to a resident, and `package` stores information regarding when packages are received and subsequently delivered. The `mailbox` table contains every mailbox and its associated key. Finally, the `special_groups` table contains all available special groups that a resident may be associated with.

Overall, the frontend will serve as a method of interacting with residents and rooms on the campus. There will be ten tables and seven units of functionality: look-up, roster, check-in or check-out, equipment loan, key loan, package log, and package delivery. The frontend will interact with the database by either selecting information about residents or inserting new information into various tables.

Logical Design

Campus Housing Chen ER Diagram



Physical Design and Table Contents

My table should be roughly in second normal form, as it does not have partial dependencies. All functional dependencies should depend only on a primary key. However, the `room` table makes use of the `hall_code` attribute which is not a key of the `hall` table, but is unique.

The database makes use of foreign keys through a nested structure. Because each complex contains multiple halls, and each hall contains rooms, the tables containing these entries use the unique IDs of each other to ensure that a room is within a valid hall and a hall is within a valid complex. Foreign keys are also used in `mailbox` and `assignment`.

Please view the MySQL Dump linked below to see the detailed physical design and table contents.

http://ix.cs.uoregon.edu/~slynch2/CampusHousingDB/dump/

Applications

There are many applications that interact with the campus housing database. Each application is listed below, detailing their desired effect and method of utilizing the database. Please note that each application usually displays a table of information required for that specific application. For example, in check-in and check-out, a query selects all rooms that are not currently assigned to a person (using a subquery). This allows a user to easily assign an open room to a person.

Roster: The roster uses a series of joins to display the assignment of all current residents. Because an assignment contains the room_id of a room, a room is within a hall, and a hall is within a complex, we use the respective keys of each to retrieve all information about an assignment. Note that the roster only shows current assignments and does not display any assignment with an `out_date` that is not null.

Look-up: This uses a slightly altered query from the roster. Lookup will retrieve only a single entry of information about a person using only their name. This would be useful for largely populated `assignment` and `resident` tables where looking through the roster for a single person would be very difficult. It assumes that there are no duplicate names of residents.

Check-In and Check-Out: These have the most complicated queries. When a new resident is entered, they are first added to the `resident` table. Then, a select query retrieves their unique `resident_id` and stores it as a variable. A second select query retrieves the unique `room_id` of the room entered and stores it as a variable. Finally, the unique `room_id` and `resident_id` are inserted into the `assignment` table as a new entry with the current date. When checking out a

resident. Their unique `resident_id` is selected and stored, then their current assignment is updated with an `out_date` of the current time and day.

Equipment Loan: Selects and displays a table of all equipment without an `in_date`. Any item and ID can be entered, logging that item as loaned out to that ID number by inserting a new entry in the `equip_borrow` table.

Equipment Return: A query selects all entries in `equipment_borrow`. A user can enter an item and ID from that table where there is no return date, then a query will update that row with the current date as the return date.

Key Loan: A query selects and displays all entries in `key_borrow` where there is no `in_date`. A separate query selects and displays all current residents along with their respective rooms and room key codes. When a user enters a resident's ID, a query selects their unique key code and saves it as a variable. This is then inserted in the `key_borrow` table with the resident's ID and the current date as the `out_date`.

Key Return: A query selects and displays all entries in `key_borrow`. If a user enters the ID of a student who has an entry in `key_borrow` without a return date, then entry is updated with the current date as the return date. It assumes no duplicate entries.

Package Log: A query selects and displays all rows in `package` without a set `out_date`, indicating all packages that have not been picked up. A user can enter a name, ID, and tracking number to log a new package which will insert the information into the `package` table.

Package Deliver: A query selects and displays all entries in `package`. If a user enters the ID and tracking number of an entry without an `in_date`, the package will be delivered by setting the `in_date` to today.

User Guide

The home page of the system provides several tools to interact with the Campus Housing database. There are four categories of tools: administrative, equipment, keys, and packages. Each page will provide a form that the user can enter information into to either select, insert, or update information in the database. The database can be accessed with the following information:

Server: "ix.cs.uoregon.edu"

Database: "housing"

User: "guest"

Pass: "guest"

Port: "3240"

Administrative Tools –

Roster: Clicking on the "Roster" button from the homepage will bring the user to a page consisting of the current roster of residents. There is no form of interaction on this page, but its information is updated based on the effects of other tools. It displays all current campus housing assignments by listing their name, hall, room, complex, mailbox, and any group they are a part of.

Look-Up: To find a current resident, enter their complete first and last name in the form and press submit. This will provide their name, hall, room, and complex. A list of all available names to search can be found from the roster.

Check-In and Check-Out: To check-in a new resident, enter an ID number, first and last name, an optional group code, a gender, and optional mailbox number. Then, from the table of open rooms listed at the bottom of the page, enter a complex name, hall name, and room number. Pressing submit will create a resident entry for the person and assign them to the room you entered. The room will be removed from the table below and the person will have an entry on the roster page.

To check-out a resident, enter their student ID, first, and last name. They will be removed from the roster page and the newly available room will appear on the list of rooms below.

Equipment –

Check-Out: Enter the name of an item and a resident's ID number, then press submit. This will check-out that item to the resident and the interaction will appear in the list below of equipment currently checked out.

Check-In: Enter the name of an item and a resident's ID number from the list of equipment currently checked out below, then press submit. The equipment will be returned by updating the log with a return date, which can be seen in the table below.

Keys –

Check-Out: To check out a key to a resident, enter their student ID number and press submit. All current residents along with their IDs and key versions are listed at the bottom of the page. They will then appear in the "Keys Currently Out" table with their respective ID and key code.

Check-In: To check-in a key, enter a resident's ID from the table of key checkouts below where the "In Date" is empty. Pressing submit will add the current date and time to the "In Date" which indicates that their key has been returned.

Packages –

Log: To log a package, enter a recipient name, ID number, and a tracking number. Pressing submit will log the package in the package list, noting the date it was received.

Deliver: To deliver a previously logged package, enter the student's ID and the unique package tracking number of a package listed below without an "Out-Date." Pressing submit will update the "Out-Date" value to the current time and date.

Conclusion

I have created a system of interactions like what actual universities use to maintain campus housing. The database stores all information about residents, housing assignments, all rooms, halls, and complexes on campus, and logs items of interaction with residents through loaning keys, loaning equipment, and delivering packages. My implementation is on a much smaller scale, but serves the same purpose.

With more time, I would be more careful with my implementation. Currently, I assume that duplicate entries will not be added to the database as I do not check for any when inserting new entries or updating information based solely on a person's name. I would also add buttons to deliver packages or return items specific to an entry in a list rather than have the user type in the information required to ensure that the desired entry is being altered. I would add more functionality for special groups and mailboxes, as they currently do not serve much of a purpose.