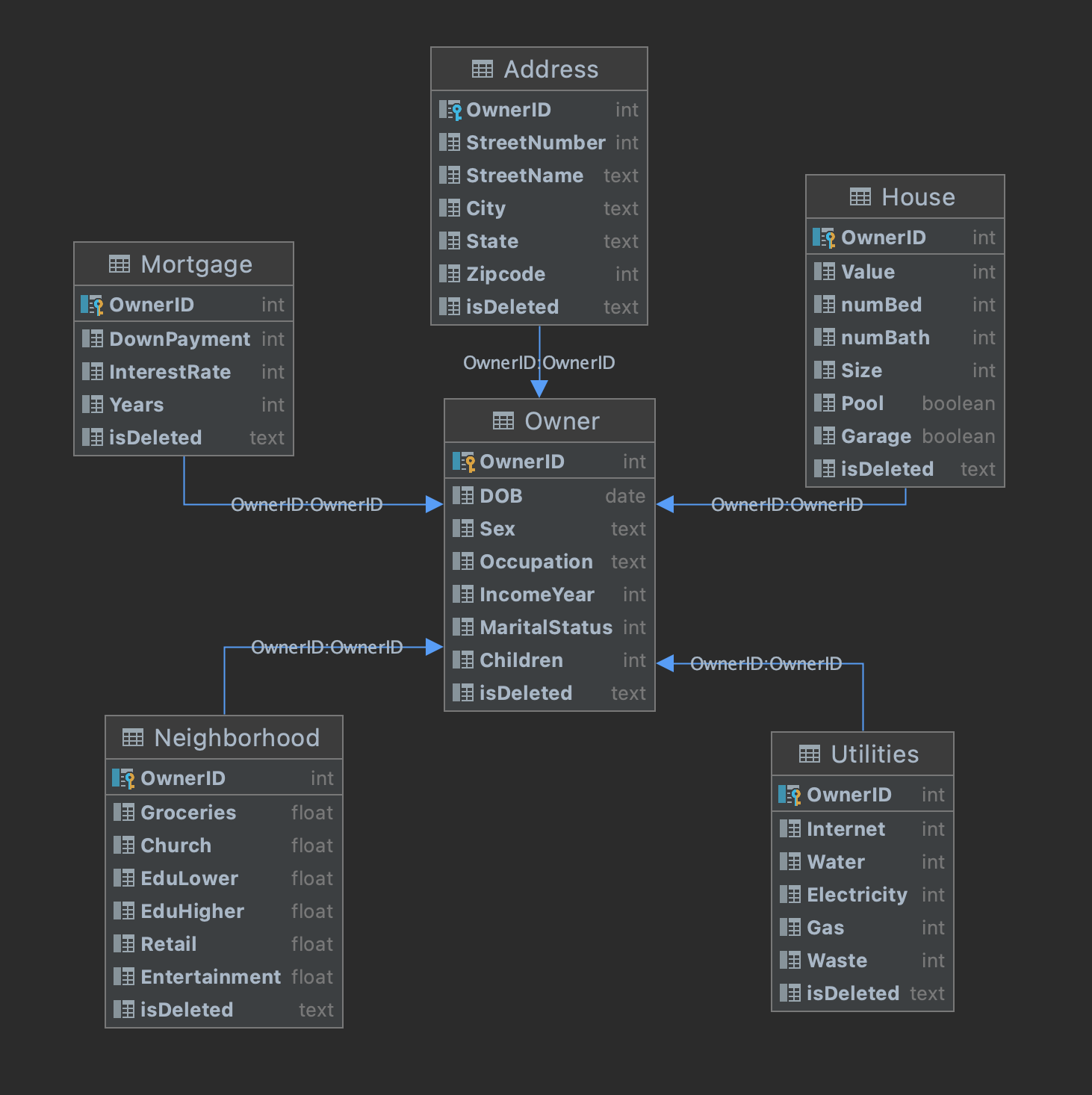
Aleksei Furlong and Saman Kashanchi

FindEstate

Searching for the right home is a long and arduous process that requires meticulous planning and patience. The research involved in this process includes contacting banks and realtors, searching for the right location and financial planning. All of this includes searching various websites and contacting various people which is why it is not only difficult for the buyer/renter but also for the real estate agent and mortgage bank. Real Estate compared to other industries lacks technical integration of the resources used in the process and the applications that do provide this have not enough information. Zillow is an upcoming platform on the market but it is limited to providing basic property information for the buyer which still requires potential customers to search elsewhere leaving room for potential applications.

Our solution to this problem is our database project called FindEstate. This database provides all relevant information of the owner and property in order to alleviate all problems concerning house searching. This allows the database not only services for potential buyers but also aids real estate agents and mortgage officers. A centralized database in which all can access will provide transparency that can make the process more efficient and allow for the potential of new services to occur. These services can be created mainly by companies who need real estate information to find potential clients which will be possible due to this database. The goal is to make FindEstate a real estate “Wikipedia” that is easy to access yet also provides a variety of tools to aid such as personalized filtering queries and data visualisation.

FindEstate consists of 7 tables that each incorporate specific information of an owner and it’s property. The primary key for this database is OwnerID and is connected to each of the tables: Address, House , Mortgage , Neighborhood, Owner and Utilities. The first table is Address and contains the street number, street name, city , state and zipcode of the house associated with a specific owner. The next table is called House which contains the price value of the home, number of bedrooms and bathrooms, size and whether the house has a pool and garage on the property. The mortgage table contained downpayment, interest rate and length of the mortgage associated with the owner. Neighborhood contains the distance in miles to the nearest Groceries, Church, lower school, upper school, retail and entertainment locations. The Owner table has information on the owner’s date of birth, sex, annual income, marital status and the amount of children they have. The last table Utilities include costs such as internet, water,electricity,gas and waste. The primary key is OwnerID in the Owner table and it’s foreign keys are the OwnerID variables that are located in each of the other tables.



RealEstate opens with a general menu that asks the user to input whether they would like to “manage ID” or try the “ personalized features”. The “Manage ID” sections represent the queries needed for Search, Update, Add and Delete. In the search function, another menu pops that allows the user to search information based on each table by entering the OwnerID. The Update function is where the user can update the Owner information or the house information by inputting the new values in each table's respective column. The Add function allows the user to add a new OwnerID based on each of the tables information allowing to not add all the information associated with the OwnerID at the same time. Lastly the delete function performs a soft delete where instead of deleting the OwnerID of the database altogether, each table will have an “isDeleted” column that has a boolean value of 1 if that OwnerID and its row of data is considered “deleted”. Each of the functions have a specific menu in which the user can visually see the progression of the interface as well as a “go back” option that provides easy navigation.

The “Personalized feature” option provides three specialized queries that users can use: filtration, graphical representation and demographic insights. The filtration feature is designed for the user to enter certain conditions that will provide a list of properties that fit. These conditions include three choices that one can filter: Budget, Location,Home Layout and compare home. With the budget condition, the user will enter a minimum and maximum value of the property. The location filter allows the user to list properties by city or state. Lastly the Home Layout allows the user to choose the number of bedrooms and bathrooms, minimum and maximum size in square feet, and whether the user would like a pool and/or garage. Using the matplotlib module, the database was able to provide a graphical representation of the data. Within the graphical representation feature, there are two options which are comparing value vs size of the property and value vs the number of bedrooms in the property. The value vs size plot was a scatterplot where one can see the individual properties represented as data points while value vs number of bedrooms is a bar plot that represents the price by grouping properties based on their bedrooms. The last specialized query was the demographics feature. This feature provided financial information based on city or state. Within the financial information was shown the average property value, income by city/state.

Unfortunately for our database we used the faker module to generate the database so the results are not meaningful. However we were effectively organising data efficiently and logically whilst providing important information special features that a user would like to use. With the layout and tools provided, the user will only need a few preferences and the queries in place will be able to solve the user’s solutions.

The current version of FindEstate is able to collect a large variety of data and handle numerous important queries however there is my work to be done. For the future, we would like to input real data to test the effectiveness of queries such as the graphical representation. We would also like to add more queries to provide users with more tools. For example we want to enhance our visual representation of data by providing satellite imagery of house and neighborhood information. Once we add real data and extra query, our back-end capabilities are acceptable to start providing a front end user website interface with proper visuals.