Plantly Database Manager

Describing the database

I created a database for an app I want to design called 'Plantly.' The goal is to manage users, users' plants, information on said plants, and groups of plants. Before describing it, it would be good to summarize the Plantly app. Plantly is a pet cam for your plants. Instead of a video of your plant (which probably is not helpful), Plantly will read back a real-time health status of your plants. Therefore, Plantly needs to know its users, the plants' users have registered, and of course, information on those plants. It would be good to disclaim that the 'pet cam' of all of this is a sensor in your plant pot sending information in real-time to your plant. The project's scope does not include relations to support the sensor. Therefore, the ER diagram below is a database manager to accommodate information on the users and the plants. Again, the goal is to add and remove the plants' and users' data and look up details on the plants.

Starting with 'Users' going clockwise,' I will describe each relation ad its attributes. The Users relation contains the information for the user to log in and for the system to track the user. It has a one-to-one relationship with 'registered_plants.' The following relation, 'registered_plants,' contains information on the user's registered plants in the app. The monitored plants with the sensor are in this table.

The following relation, 'Plant_status,' holds the actual status of those registered plants. The Plant_Status table has data on the condition of the plant and not on how the plant *should* be doing. So far, we are managing information on the users, the plants registered to those users, and the current conditions. The following relation provides information on what a 'healthy plant' is. The 'PlantCatalog' is a dictionary on plants and their ideal conditions. To know if a plant lacks or receives too much of something, the user needs to know the ideal amount. The right conditions tell the user what they need to change.

Finally, the 'Groups' category is a one-to-many on the PlantCatalog side and a many-to-many on the registered_plants side. There can be one plant type in multiple groups, and many registered plants will be in numerous groups. The group's relation categorizes plants into clusters of similar attributes. The application does not need to pull every plant's detail. Instead, it can get the group and other plants in that group. The application did not use the 'groups' relation as it was outside the scope.

Grading Check List

Tables used used for getting:
1) users 2) registered_plants 3) PlantCatalog
Tables used for setting:
1) users
Join operation:

1) The application combines 'users' and 'registered_plants' to show each user's registered plants. Note! Not all users have plants registered to them. Choose the first user for a better demo. One more note, the users 'listbox' will display the same user's name if you

try to add more than one user per session. If you rerun the application, the new users will show that the application is setting the database.

Plantly ER Diagram

carson laudadio | March 15, 2022