TCSS 445

Petter

Project Phase II

Weiwei Shi

Vinh Vien

(Data Requirements. List all the data requirements for your system. These requirements are what you used to build your ER diagram. You may have revised your requirements from the proposal to capture details about the system. Even if you listed requirements in the proposal, create a separate document that lists them. We will do some examples in class to help you with this.)

SUMMARY & LIST OF CHANGES:

● We were planning to make a post wall for the pets, similar to a facebook wall. This was voted down because we want the website to only focus on finding pets.

● We will implement a table about ratings of pets. Rating are based of off the encounter with the pets.

● A messaging option will be added to our website to help user contact one another. This is to substitute the 'facebook' portion of the project which was removed.

● Originally the AccountType bridge table was for merging a many to many relationship between User and Account. However as we build the tables we realize it was just a one to many relationship. Instead we change the Table Type to a table representing the type of account a user can have. This is so future updates can create new account without changing structure of database.

Ideas worded:

A user can have many accounts.

An account can only have one user.

An account have multiple pets.

A pet can only link to one account.

A pet can have multiple rating.

A rating only applies to one pet.

An account can have multiple messages; Sent and Received.

A message can only relate to one account. Either Sender or Receiver.

A TypeID link to a table to represent the type name/title for the account.

User Table

● <u>UserID</u>: int. AUTO_INCREAMENT acctID will be the primary key for the Account Table. Also foreign key for Account table.

● Email: varChar(45). Email will be used as User's login username.

● Password: varChar(15). User's password.

● UserCreatedOn: date. Store the User's create the account date.

Type Table

Purpose of this table to keep track of the title of the account. The benefit of doing this over an enum is that updating account type in the future will not cause me to change the structure of the database.

● <u>TypeID</u>: int. The primary key for this table, integer that represent account type. This is foreign key for Account table.

● TypeName: varChar(15). The title of the account. Example: Owner, Adoption center, Breeder, and Regular users.

Account Table

● <u>UserID</u>: int. UserID is the primary key for the Account Table. Also a foreign key for User table.

● <u>TypeID</u>: int. Another primary key for Account Table and also foreign key for Type table. The number that represents the account type.

● FName: varChar(15). User's first name.

● LName: varChar(15). User's last name.

● PhoneNum: varChar(11). User's phone number.

● Gender: varChar(1). User's gender, 'F' or 'M'

● Street: varChar(45). User's street information.

● City: varChar(25). User's City.

● State: varChar(2). User's State

● Zipcode: varChar(10). User's Zip code.

Pet Table

● PetID: int. Pet's ID, it's primary key for the Pet Table. Also foreign key to Rating table.

● OwnerID: int. The user the pet belong too. Also foreign key to Account table.

● Name: varChar(15). Pet's name.

● PetType: varChar(15). Pet's type, dog, cat, bird, etc,.

● Breed: varChar(15). Pet's breed, husky, teddy, etc,.

● Age: int. Pet's age.

● Sex: varChar(1). Pet's sex, 'F' or 'M'

● Color: varChar(10). Pet's color.

● Size: int. Pet's size Depending on the pet the interface will determine size in a different way.

● City: varChar(25). Stores which city the pet lives.

● State: varChar(2). Stores which state the pet lives.

● Hobby: varChar(100). List pet's hobbies.

● Image: blob. Image of pet

Rating Table

● RateID: int. The number that represent the rating post It's the primary key for the table

● PetID: int. The pet that is being rated. Also foreign key for Pet Table. Categories of ratings.

● Cuteness: int. Only accept numbers 1 to 5

● Softness: int. Only accept numbers 1 to 5

● Hairy: int. Only accept numbers 1 to 5

● Healthy: int. Only accept numbers 1 to 5

● Clean: varchar(1). Accepts 'y' or 'n'. Representing yes or no.

● Happy: int. Only accept numbers 1 to 5

● Energetic: int.Only accept numbers 1 to 5

● Overall Experience: int. Only accept numbers 1 to 5

● Comments: mediumText. Any comments the user wants to add.

Messages Table

- <u>SenderUserID</u>: int. The sender of the message
- <u>ReceiverUserID</u>: int. The receiver of the message
- Date: DateTime. The date and time message sent
- Message: mediumText. The message sent