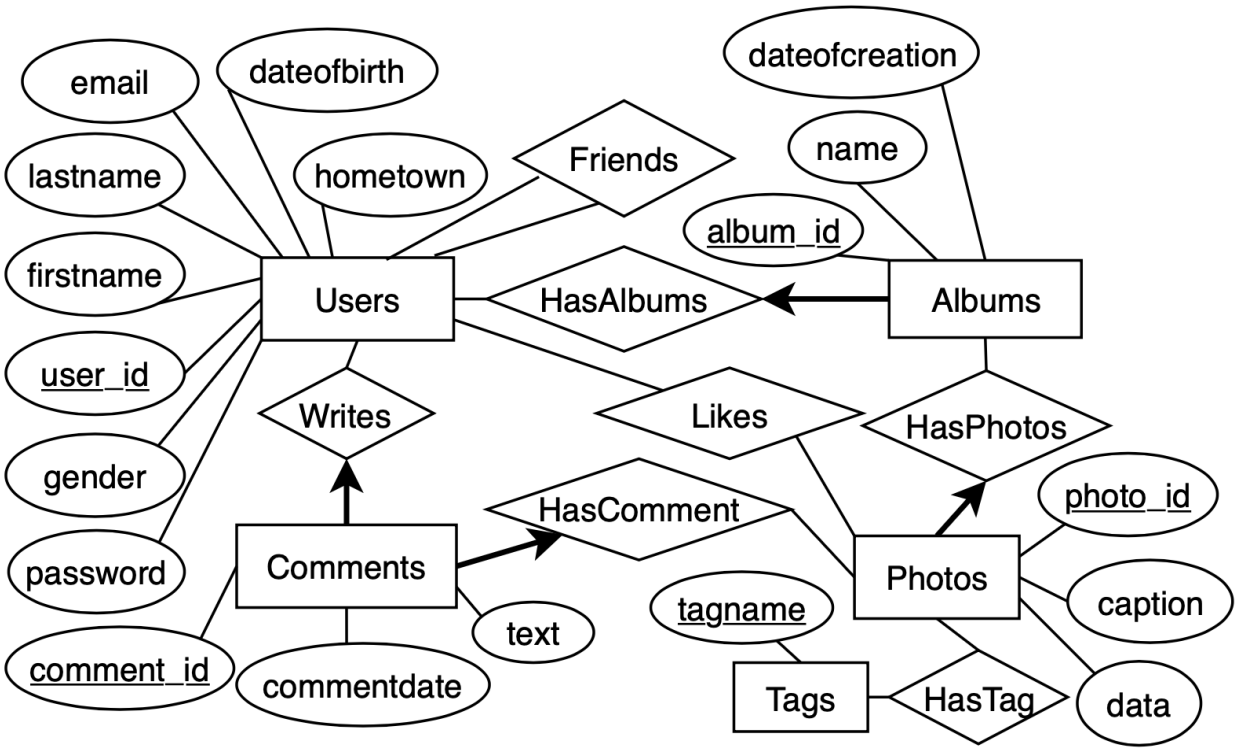
Entity Relationship Diagram



Relational Model Schema

1. Users (user_id, email, password, firstName, lastName, hometown, gender, dateOfBirth)
2. Friends (user_id, friend_id)
3. Albums (album_id, name, dateofcreation, user_id)
4. Photos (photo_id, data, caption, album_id)
5. Tags (tagname, photo_id)
6. Comments (comment_id, commentdate, text, user_id, photo_id)
7. Likes (photo_id, user_id)

Integrity Constraints & Assumptions

1. An album is created by exactly one user, a photo is a part of exactly one album, a comment is written by exactly one user for exactly one photo and a like is made by exactly one user for exactly one photo and so on. All these relationships are one-many and have total participation constraints using foreign key constraints etc.

2. Primary key constraints are defined for each table there is. There are unique key constraints wherever necessary such as for user's email and the album name for each user i.e. no two (or more) users can have same email addresses & no user can have two (or more) albums with the same album name.

3. Constraints are defined on user creation to ensure that a password is strong with >= 8 characters and to validate email addresses as much as possible.

4. If a user is deleted, all associated albums will be deleted automatically, and thus on deletion of an album all associated photos will be deleted, and on deletion of a photo, all associated comments and likes will be deleted. However, is a user is deleted, if they have a comment/like on another user's photo, that comment/like shall not be deleted.

What we could do further? (delete this section before submitting)

1. Add TRIGGER CONSTRAINT for Tags that prevents more than 5 tags to be associated per photo_id
2. Add a constraint where if a photo_id is deleted, the tags associated do not get deleted and at least one instance of each unique tag remains.
3. Add TRIGGER CONSTRAINT for Friends that automatically creates the reverse relation and adds it to the table. Example: if (54, 67) is added to Table Friends, the Trigger should check if reverse i.e. (67, 54) exists, and if it does not exist, it should add (67, 54) to the Table.
4. Add CHECK statement to check if dateofbirth is less than currentdate and greater than say year 1900

SQL Statements

```
CREATE DATABASE photoshare;
USE photoshare;

CREATE TABLE Users (
    user_id INT NOT NULL AUTO_INCREMENT UNIQUE PRIMARY KEY,
    email VARCHAR(255) NOT NULL UNIQUE,
    password VARCHAR(255) NOT NULL,
    firstName VARCHAR(255) NOT NULL,
    lastName VARCHAR(255),
    hometown VARCHAR(255),
    gender ENUM('M', 'F'),
    dateOfBirth DATE NOT NULL,

    CONSTRAINT user_email_validation
                CHECK (email REGEXP
"^[a-zA-Z0-9][a-zA-Z0-9.!#$%&'*+-/=?^_`{|}~]*?[a-zA-Z0-9._-]?@[a-zA-Z0-9][a-zA-Z0-9._-]*?[a-z
A-Z0-9]?\\.[a-zA-Z]{2,63}$"),

CONSTRAINT user_password_is_strong
                CHECK (password LIKE '%[0-9]%' AND password LIKE '%[A-Z]%' AND
password LIKE '%[a-z]%' AND LENGTH(password) >= 8)

);

CREATE TABLE Friends (
 user_id INT NOT NULL,
 friend_id INT NOT NULL,
 PRIMARY KEY  (user_id, friend_id),
 FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE,
 FOREIGN KEY (friend_id) REFERENCES Users(user_id) ON DELETE CASCADE
);

CREATE TABLE Albums (
  album_id INT NOT NULL AUTO_INCREMENT UNIQUE PRIMARY KEY,
  name VARCHAR(255) NOT NULL,
  date_created DATE NOT NULL,
  user_id INT NOT NULL,
  FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE,
  UNIQUE KEY albumid (user_id, name)
);
```

```
CREATE TABLE Photos (
  photo_id INT AUTO_INCREMENT UNIQUE PRIMARY KEY,
  data VARCHAR(255) NOT NULL,
  caption VARCHAR(255) NOT NULL,
  album_id INT NOT NULL,
  FOREIGN KEY (album_id) REFERENCES Albums(album_id) ON DELETE CASCADE
);

CREATE TABLE Tags (
  tagname VARCHAR(255) UNIQUE PRIMARY KEY,
  photo_id INT,
  FOREIGN KEY (photo_id) REFERENCES Photos(photo_id) ON DELETE CASCADE
);

CREATE TABLE Comments (
  comment_id INT NOT NULL AUTO_INCREMENT UNIQUE PRIMARY KEY,
  text VARCHAR(255),
  user_id INT DEFAULT '0',
  Date DATE NOT NULL,
  photo_id INT NOT NULL,
  FOREIGN KEY (user_id) REFERENCES Users(user_id),
  FOREIGN KEY (photo_id) REFERENCES Photos(photo_id) ON DELETE CASCADE
);

CREATE TABLE Likes (
  photo_id INT NOT NULL,
  user_id INT  DEFAULT '0',
  PRIMARY KEY (photo_id, user_id),
  FOREIGN KEY (photo_id) REFERENCES Photos(photo_id) ON DELETE CASCADE,
  FOREIGN KEY (user_id) REFERENCES Users(user_id)
);
```