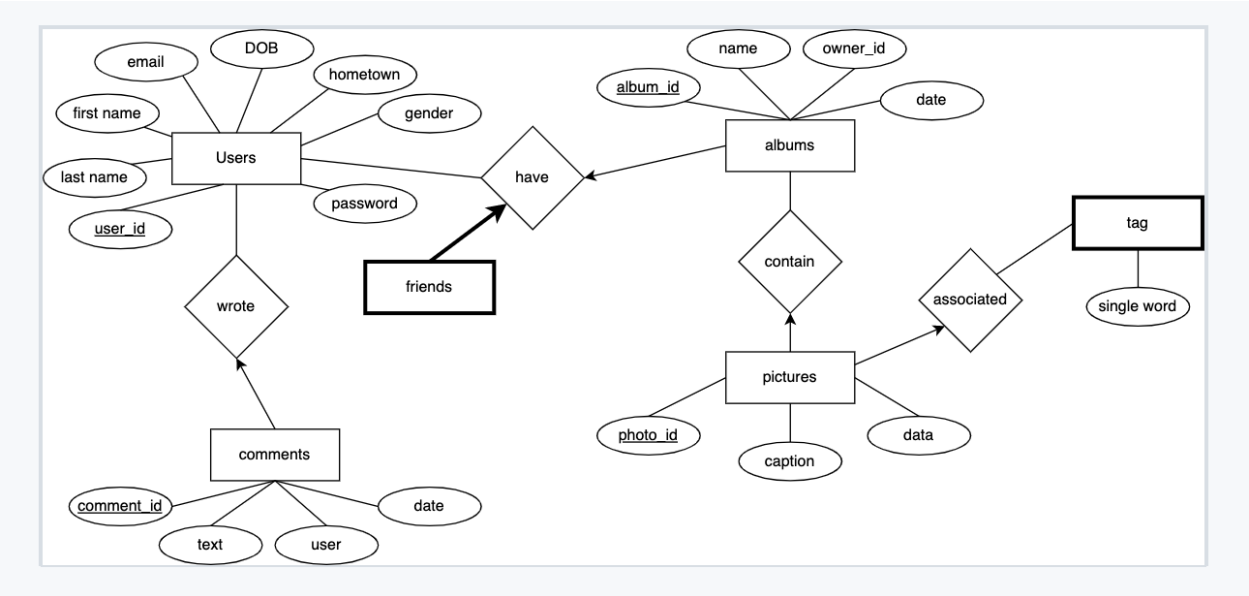
Group members: Youngjin Shin, Sam Leach

ER DIAGRAM



RELATIONAL SCHEMA

/* Youngjin Shin & Sam Leach */

```
CREATE DATABASE IF NOT EXISTS photoshare;
USE photoshare;

DROP TABLE IF EXISTS Users CASCADE;
DROP TABLE IF EXISTS Pictures CASCADE;
DROP TABLE IF EXISTS Friends CASCADE;
DROP TABLE IF EXISTS Albums CASCADE;
DROP TABLE IF EXISTS Comments CASCADE;
DROP TABLE IF EXISTS Tags CASCADE;

CREATE TABLE Users (
   user_id int4  AUTO_INCREMENT,
   email varchar(255) UNIQUE,
   user_pass varchar(255),
   firstname varchar(255),
   lastname varchar(255),
   dob date,
   hometown varchar(255),
   gender varchar(30),
```

```
  CONSTRAINT users_pk PRIMARY KEY (user_id)
);

CREATE TABLE Pictures
(
  picture_id int4  AUTO_INCREMENT,
  user_id int4,
  imgdata longblob ,
  caption VARCHAR(255),
  album_id int4,
  INDEX upid_idx (user_id),
  CONSTRAINT album_fk FOREIGN KEY (album_id) references Albums(albums_id),
  CONSTRAINT pictures_pk PRIMARY KEY (picture_id)
);

CREATE TABLE Friends(
   new_friend_id int4,
   follower_id int4,
CONSTRAINT following_fk FOREIGN KEY (following_id) REFERENCES Users(user_id),
CONSTRAINT follower_fk FOREIGN KEY (follower_id) REFERENCES Users(user_id)
);

CREATE TABLE Albums(
          albums_id int4 AUTO_INCREMENT,
          album_name varchar(50),
          albumDate date,
    owner_id int4,
CONSTRAINT albums_pk PRIMARY KEY(albums_id),
CONSTRAINT owner_fk FOREIGN KEY (owner_id) REFERENCES Users(user_id)
);

CREATE TABLE Comments(
          comments_id int4 AUTO_INCREMENT,
    comment_owner int4 NOT NULL,
    comment_text varchar(255),
    commentDate date,
CONSTRAINT comments_pk PRIMARY KEY(comments_id),
CONSTRAINT comment_owner_fk FOREIGN KEY (comment_owner) REFERENCES Users(user_id)
);

CREATE TABLE Tags
(
  tag varchar(255),
  photo_id int4,
```

```
  CONSTRAINT tag_fk FOREIGN KEY (photo_id) REFERENCES Photos(photo_id)
);
```

ASSUMPTIONS

- Users and comments are in a many-to-one relationship.
  - Users are allowed to write multiple comments.
- Albums and pictures are in a many-to-one relationship.
  - Albums are allowed to have many pictures but each picture can only be in one album.
- Friends is a weak entity. It can only be identified by a primary key of another entity(user_id).
- Tag is a weak entity. It can only be identified by a primary key of another entity(photo_id).
  - The attribute,single word, is the partial key for tag.
- Pictures and tags are in a one-to-many relationship.
  - Since tag is a weak entity, picutures and tag must be in one-to-many relationship. A same tag can be tagged in many pictures.

INTEGRITY CONSTRAINTS

- the user_id in the Users table is a primary key to have a unique identifier for each user
- the picture_id in the Pictures table is a primary key to have a unique identifier for each picture
- the album_id is a foreign key in the Pictures table that references the albums_id attribute in the Albums table
- their is a participation constraint established in the pictures table where each picture can only have one Album assigned to it using the foreign key mentioned above
- their are two foreign keys in the Friends table, new_friend_id and follower_id that each reference the user_id of the person who sent the friend request and the person who recieved it
- the albums_id in the Albums table is the primary key to give a unique identifier to each album
- the owner_id in the Albums table is a foreign key that references the user_id of the creater of said album
- the comments_id attribute in the Comments table is the primary key to give a unique identifier to each comment
- the comment_owner attribute in the Comments table is a foreign key that references the user_id of the person who posted the comment
- the photo_id attribute in the Tags table is a foreign key that references the photo_id of the photo the tag is assigned to