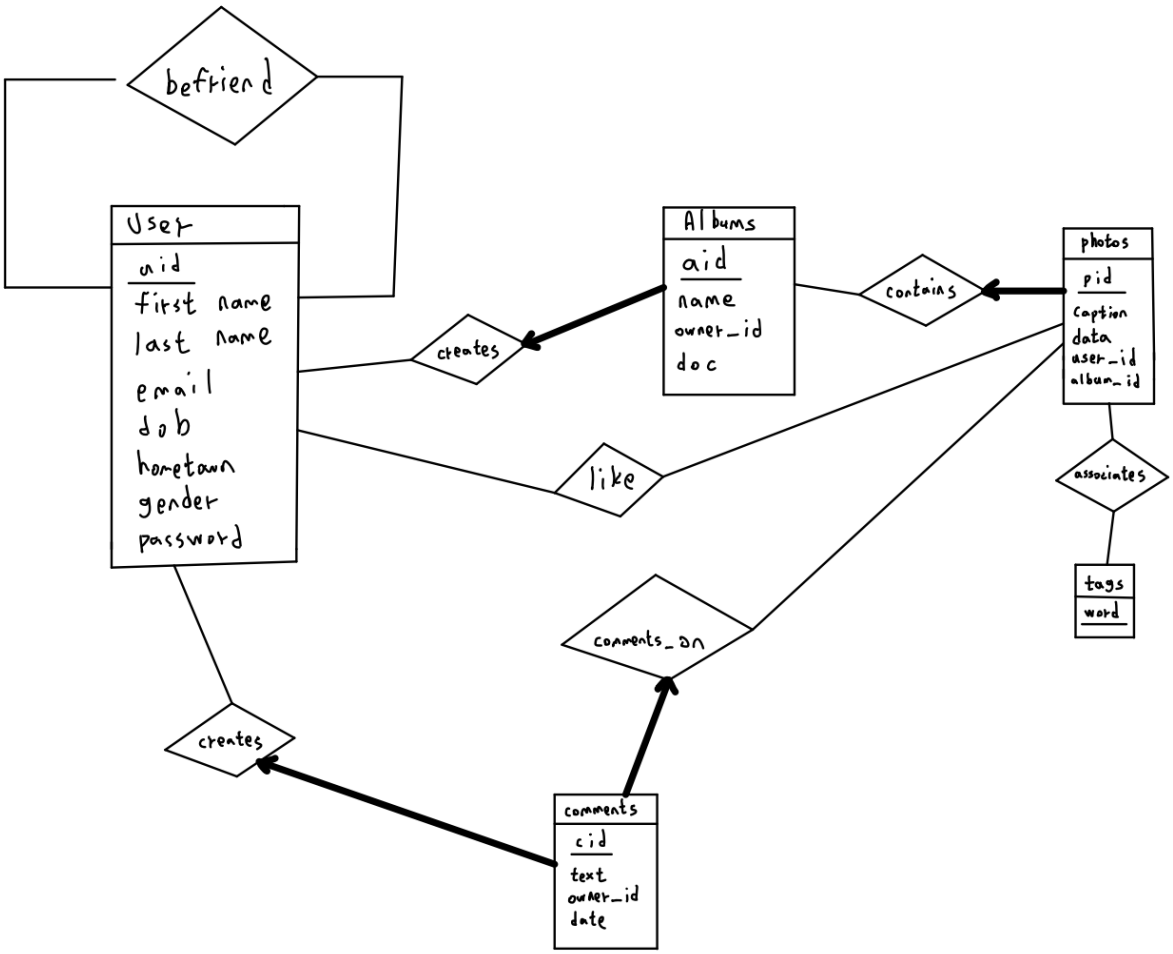

Additional Assumptions:

1. “Friend” functions more like “follow” — it does not need permission for user A to add user B as friend.
2. Each album must have exactly one owner(creator).
3. Each photo must belong to exactly one album (A user has to create an album before that user can upload a photo).
4. Each comment (identified by comment_id) can only be created by one user, to comment on one photo (but different comments may have the same content).
5. User uses email to register, so email must be unique.
6. Users cannot befriend with themselves.
7. Users cannot comment on their own photos.

In the SQL below, some “exactly one” restriction is translated by using attributes. For example, since one album must have exactly one user, we add a NOT NULL attribute named user_id. Similar for photos and comments.

```
CREATE DATABASE IF NOT exists PA1;
```

```
use PA1;
DROP TABLE IF EXISTS user_create_comment CASCADE;
DROP TABLE IF EXISTS user_like_Photo CASCADE;
DROP TABLE IF EXISTS be_friend CASCADE;
DROP TABLE IF EXISTS associate CASCADE;
DROP TABLE IF EXISTS Tags CASCADE;
DROP TABLE IF EXISTS Comments CASCADE;
DROP TABLE IF EXISTS Photos CASCADE;
DROP TABLE IF EXISTS Albums CASCADE;
DROP TABLE IF EXISTS Users CASCADE;

CREATE TABLE Users ( -- capitalized entitys for notations
    user_id INT4 AUTO_INCREMENT,
    first_name VARCHAR(20),
    last_name VARCHAR(20),
    email VARCHAR(30) UNIQUE,
    job VARCHAR(255),
    hometown VARCHAR(20),
    gender VARCHAR(20),
    password VARCHAR(255),
    CONSTRAINT users_pk PRIMARY KEY (user_id)
);

CREATE TABLE be_friend(
    user_id_from INT4,
    user_id_to INT4,
    PRIMARY KEY (user_id_from, user_id_to),
    FOREIGN KEY (user_id_to) REFERENCES Users(user_id) ON DELETE CASCADE,
    FOREIGN KEY (user_id_from) REFERENCES Users(user_id) ON DELETE
CASCADE,
    CONSTRAINT diff_user
        CHECK ((user_id_from) <> (user_id_to))
);
 -- ALTER TABLE be_friend ADD INDEX(user_id1);
 -- ALTER TABLE be_friend CHANGE user_id1 user_id1 INT4 AUTO_INCREMENT;

CREATE TABLE Albums(
    album_id INT4 PRIMARY KEY AUTO_INCREMENT,
    album_name VARCHAR(255),
    user_id INT4 NOT NULL,
    date_created date,
    FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE
);
```

```sql
CREATE TABLE Photos(
  photo_id INT4 AUTO_INCREMENT,
  user_id INT4 NOT NULL,
  album_id INT4 NOT NULL,
  imgdata LONGBLOB, -- store data in binary
  caption VARCHAR(255),
  INDEX uphoto_id_idx (user_id),
  CONSTRAINT photos_pk PRIMARY KEY (photo_id),
  FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE,
  FOREIGN KEY (album_id) REFERENCES Albums(album_id) ON DELETE CASCADE
);

CREATE TABLE Tags(
    word VARCHAR(25) PRIMARY KEY
);

CREATE TABLE associate(
    photo_id INT4,
    word VARCHAR(25),
    PRIMARY KEY (photo_id, word),
    FOREIGN KEY (photo_id) REFERENCES Photos(photo_id) ON DELETE CASCADE,
    FOREIGN KEY (word) REFERENCES Tags(word)
);


CREATE TABLE user_like_Photo(
    user_id INT4,
    photo_id INT4,
    PRIMARY KEY (user_id, photo_id),
    FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE,
    FOREIGN KEY (photo_id) REFERENCES Photos(photo_id) ON DELETE CASCADE
);

CREATE TABLE Comments(
    comment_id INT4 PRIMARY KEY AUTO_INCREMENT,
    user_id INT4 NOT NULL,
    photo_id INT4 NOT NULL,
    content VARCHAR(255),
    date_comment date,
    FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE,
    FOREIGN KEY (photo_id) REFERENCES Photos(photo_id) ON DELETE CASCADE
);
```