Each user must have an id, a first name, last name, email and password for the account to be valid. The email must be unique. The primary key is a unique user id. The other information (gender etc.) is not our priority so I didn't use NOT NULL for these. A user who does not have an account and leaves an anonymous comment will be handled by an all zeros uid.

Each album must have an albumid, a name, a user id of the user to which it belongs to. uid is a foreign key that references the user to which the album belongs to. If the user is deleted, the album should also be deleted as it would not make sense for a user to want to delete his account but still keep the albums with his pictures available for people to view on the website.

A photo must have a photoid, an albumid - which will reference the album that the photo is in. As albums have a relationship with users already, we do not need to have the uid on the photo table as a foreign key as well (because they will already be connected with the album). I assume that if the whole album is deleted, the photo will be deleted too.

A tag is a simple text that the user has written, the same tag can be written for multiple photos.

A comment must have a commentid, some text (which is the comment itself), the uid (that connects it to the user who wrote is because we want to know who left the comment), and a photoid (because the comment is on the photo). I assumed that comments are on photos, and not on albums. If the user who left the comment is deleted, the comment should also be deleted because probably he doesn't want his name associated with comments if he has wanted to delete his account- but if an anonymous user (not logged in) has commented, the comment will always stay.If the photo itself is deleted the comment is deleted because each comment it on a specific photo - so it wouldn't make sense for a comment to exist on a non-existent photo (this all zero uid will never be deleted- there will always be the case of the anonymous user which is the reason comments from anonymous users won't be deleted unless the photo on which the comment is displayed is deleted— see also user). Additionally, a constraint is placed so that the uid of the user who is commenting cannot be the same as the uid of the user that has posted the photo (although I am not sure if this is going to work in mysql but for the completeness of the design for now)

wrote describes the relationship between the user and a comment. It has as primary key both the uid and the commentid. Date should also be NOT NULL as it was written at a specific date and we want to know when.

created describes the relationship between the user and an album. It has as primary key both the uid and the albumid. Date should also be NOT NULL as it was written at a specific date and we want to know when.

commenton describes the relationship between the comment and the photo. It has as primary key both the photoid and the commentid.

has describes the relationship between a tag and a photo. So the primary key are both the text of the tag and the id of the photo.

belongs describes the relationship between a photo and an album. It has as primary key both the photoid and the albumid.

friends describes the relationship of users being friends on the website, thus between two users.

likes describes the relationship of a user “liking” a photo. The primary key is both the photoid and the uid of the one who likes it.