Databases

PhotoShr : Design and Implementation of Relational Database for Photo Sharing Social Network

Hochschule für Technik Stuttgart

University of Applied Sciences

Suman Shakya (31shsu1mst)

Nasser Abdurahman (31ahna1mst)

Summer term 2013

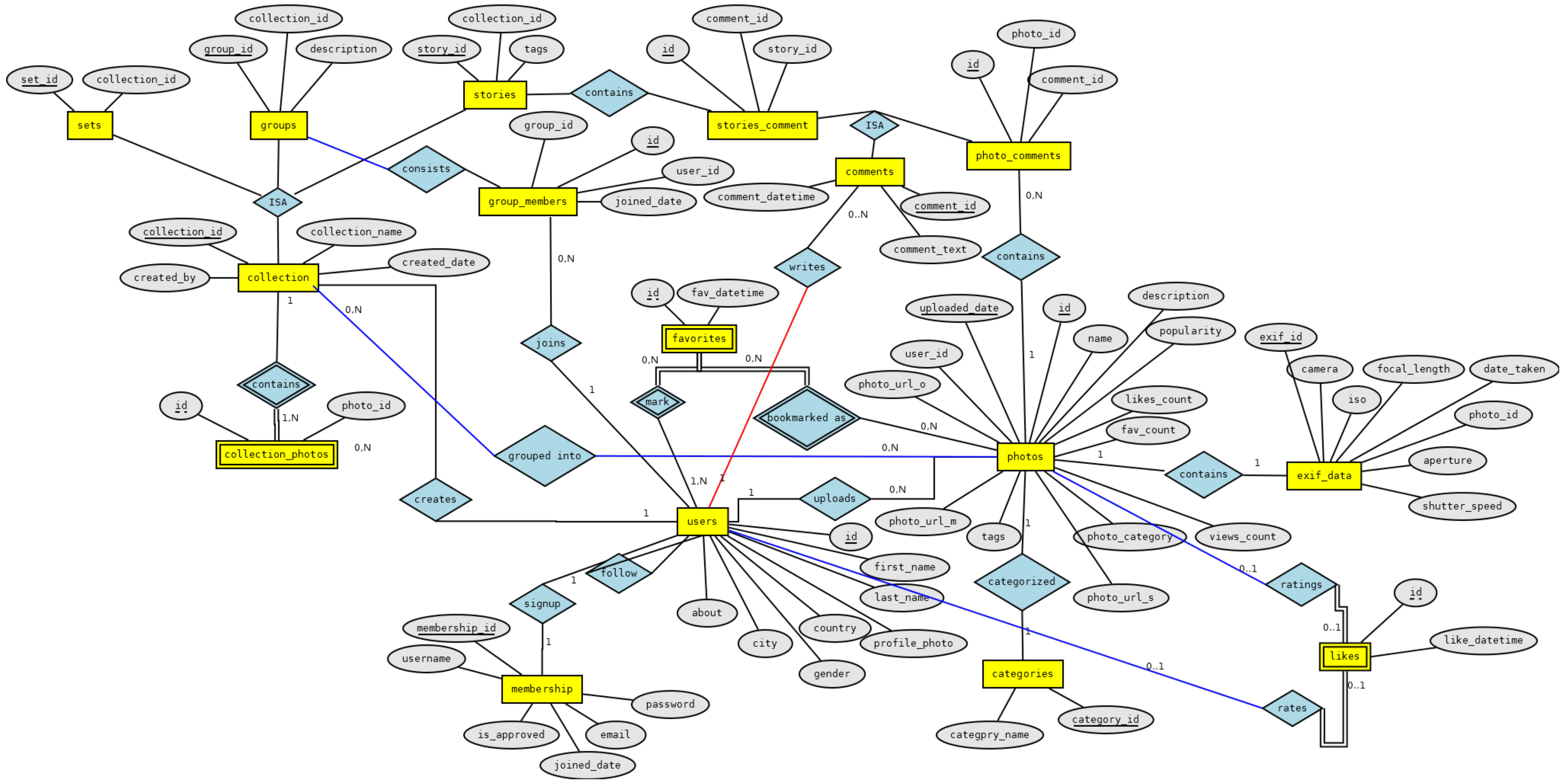
Outline

- Introduction to PhotoShr
- Database Design
  - Entity Relationship Model
  - Triggers
    - Rating Algorithm
  - Stored Procedures
- Implementation
  - Tools and Technologies Used
  - Demo
- Future Enhancement

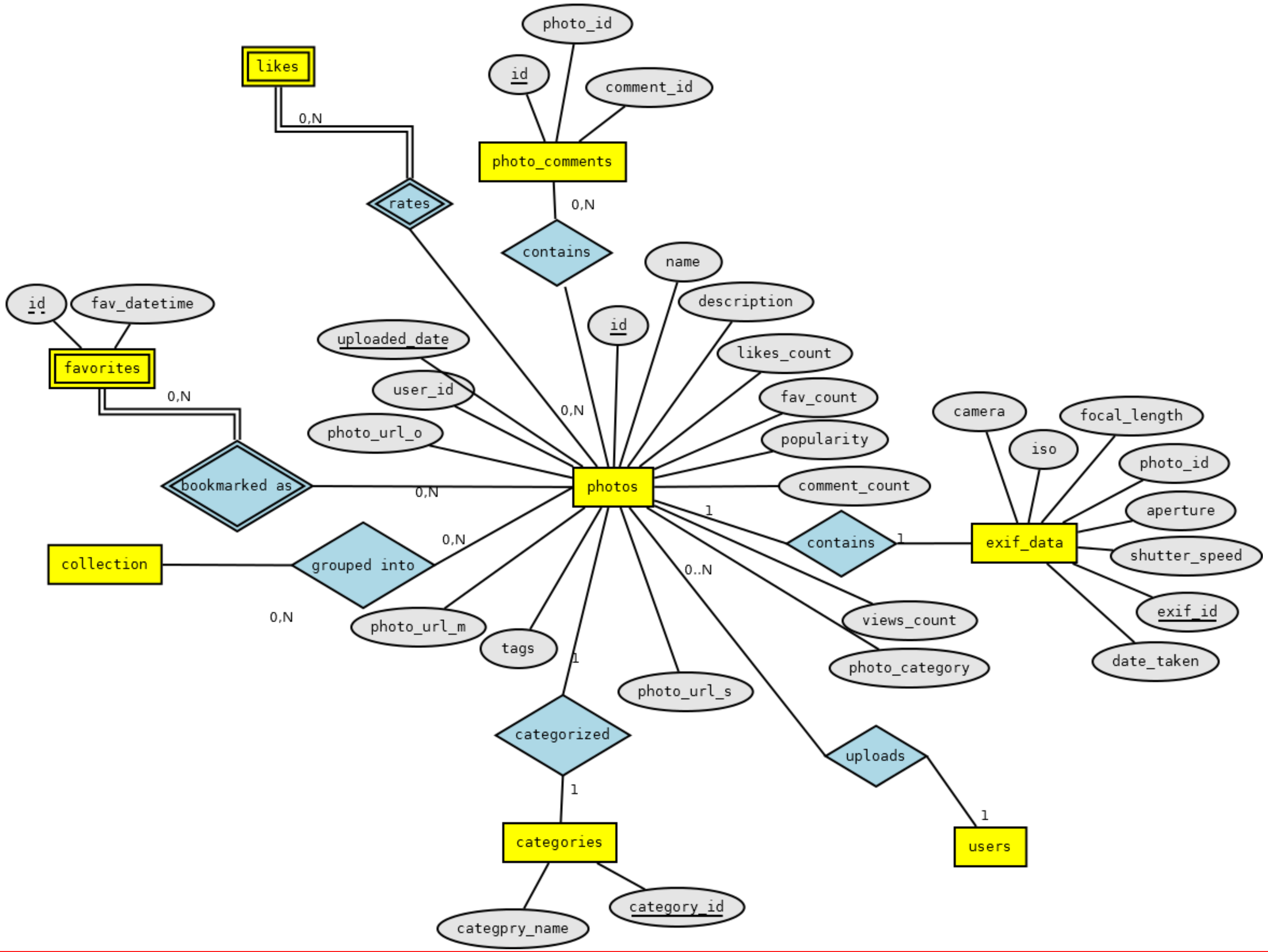
Introduction to PhotoShr

- Online Photo sharing and photo management.
- Upload photos and share with community.
- Features :
  - Add photos to favorites, rate photos by liking it.
  - Follow other people to view their content and get inspired.
  - Manage your portfolio by creating sets.
  - Write about your experience with photos by creating stories.
  - Get noticed in the community by adding photos to groups.
  - Tag photos to get your photos discovered easily.

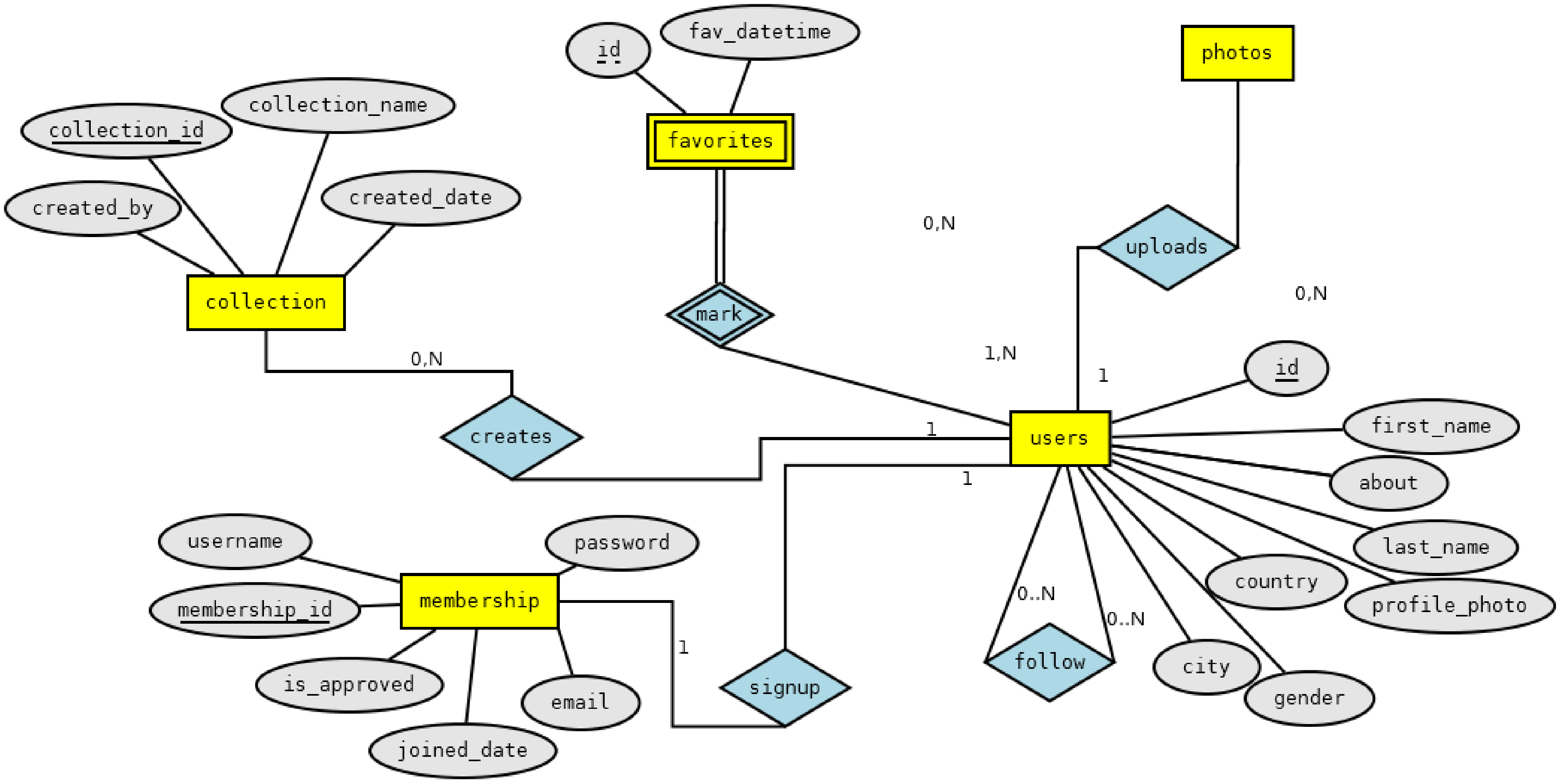
Entity RelationShip Model



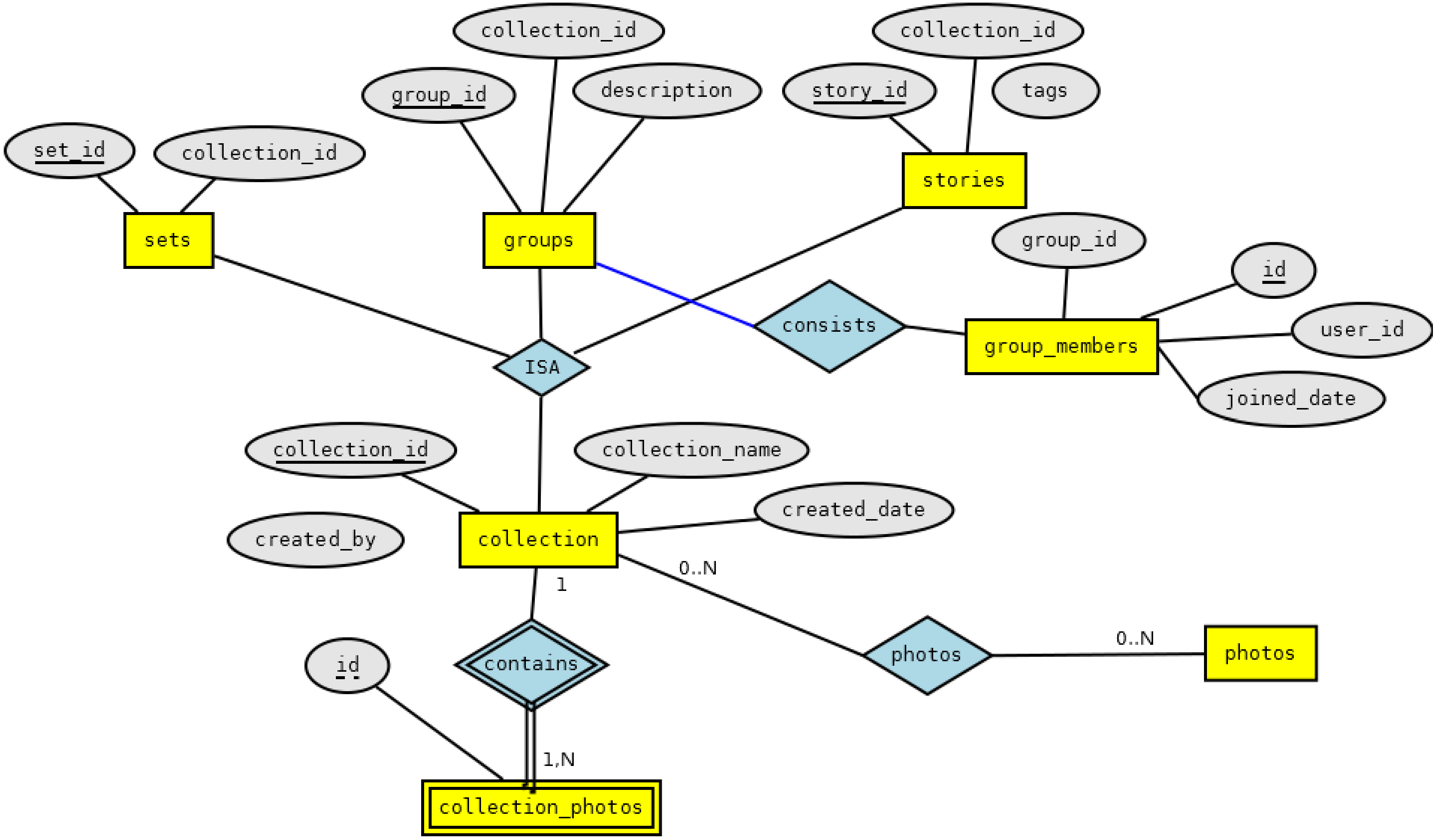
Entity RelationShip Model - Photo



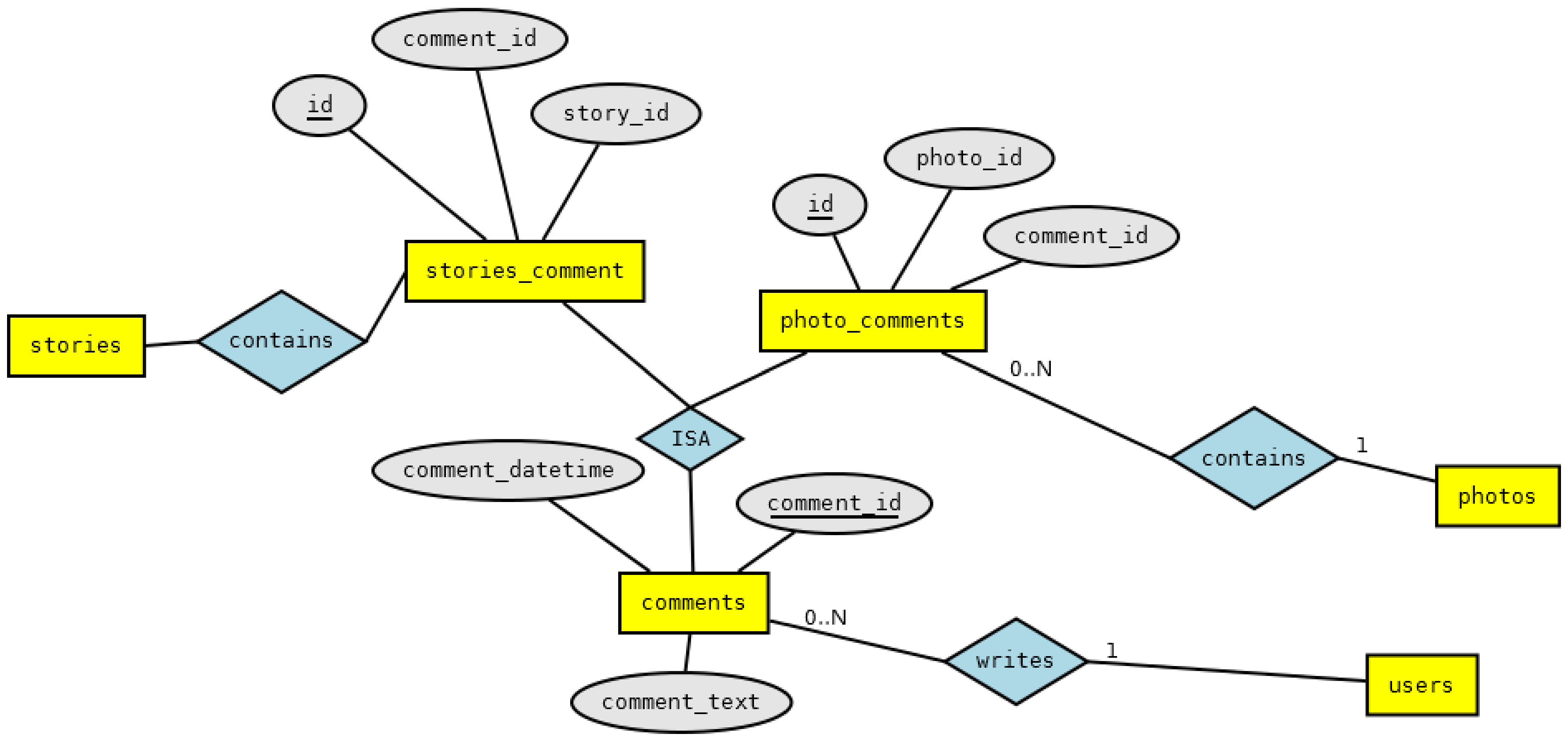
Entity RelationShip Model - Users



Entity RelationShip Model - Collection

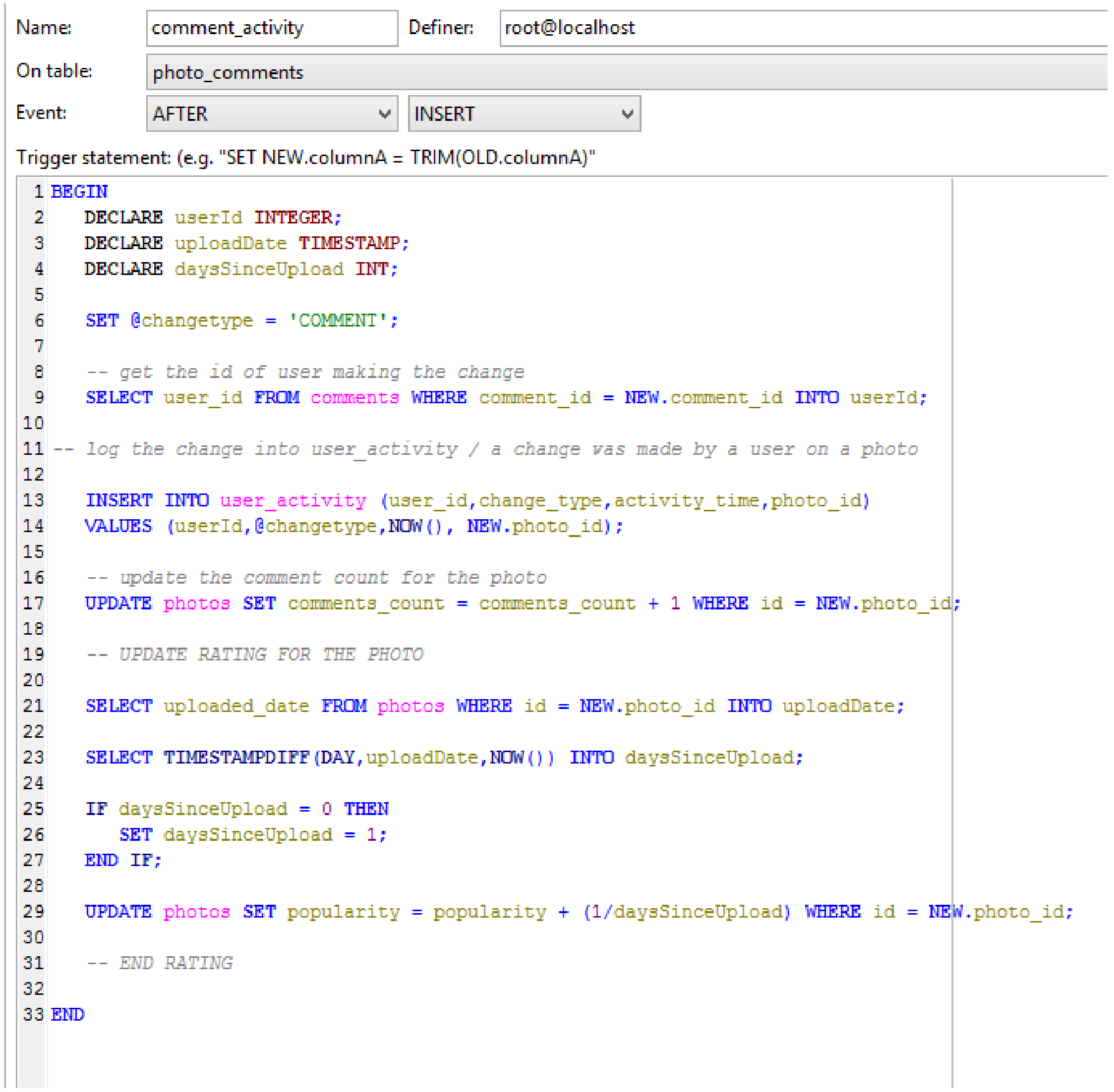


Entity RelationShip Model - Comments



Trigger

- In **PhotoShr**, triggers are used to track recent activity for a user.
- Triggers are fired when a user uploads, comments, likes or favorites a photo.

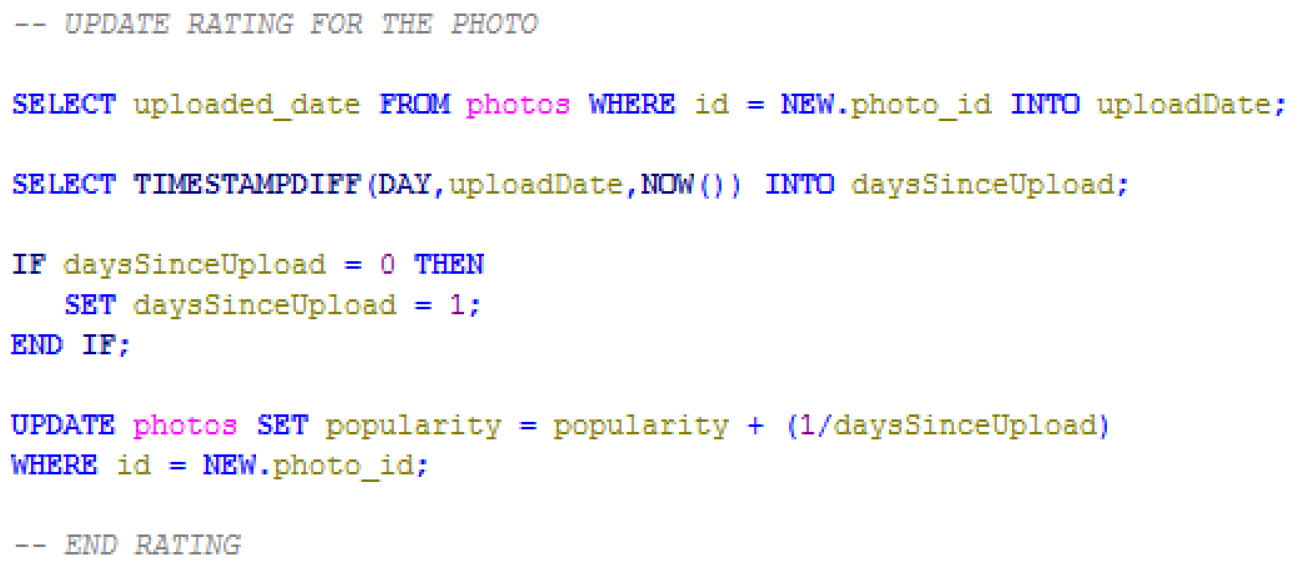

Name: comment_activity Definer: root@localhost
On table: photo_comments
Event: AFTER INSERT
Trigger statement: (e.g. "SET NEW.columnA = TRIM(OLD.columnA)"

```
BEGIN
   DECLARE userId INTEGER;
   DECLARE uploadDate TIMESTAMP;
   DECLARE daysSinceUpload INT;

   SET @changetype = 'COMMENT';

   -- get the id of user making the change
   SELECT user_id FROM comments WHERE comment_id = NEW.comment_id INTO userId;

-- log the change into user_activity / a change was made by a user on a photo

   INSERT INTO user_activity (user_id,change_type,activity_time,photo_id)
   VALUES (userId,@changetype,NOW(), NEW.photo_id);

   -- update the comment count for the photo
   UPDATE photos SET comments_count = comments_count + 1 WHERE id = NEW.photo_id;

   -- UPDATE RATING FOR THE PHOTO

   SELECT uploaded_date FROM photos WHERE id = NEW.photo_id INTO uploadDate;

   SELECT TIMESTAMPDIFF(DAY,uploadDate,NOW()) INTO daysSinceUpload;

   IF daysSinceUpload = 0 THEN
      SET daysSinceUpload = 1;
   END IF;

   UPDATE photos SET popularity = popularity + (1/daysSinceUpload) WHERE id = NEW.photo_id;

   -- END RATING

END
```

Rating Algorithm

- A very simple linear algorithm with time factor.
- Foreach (Comment/Vote/Favorite)
  - Popularity += 1/timelapse

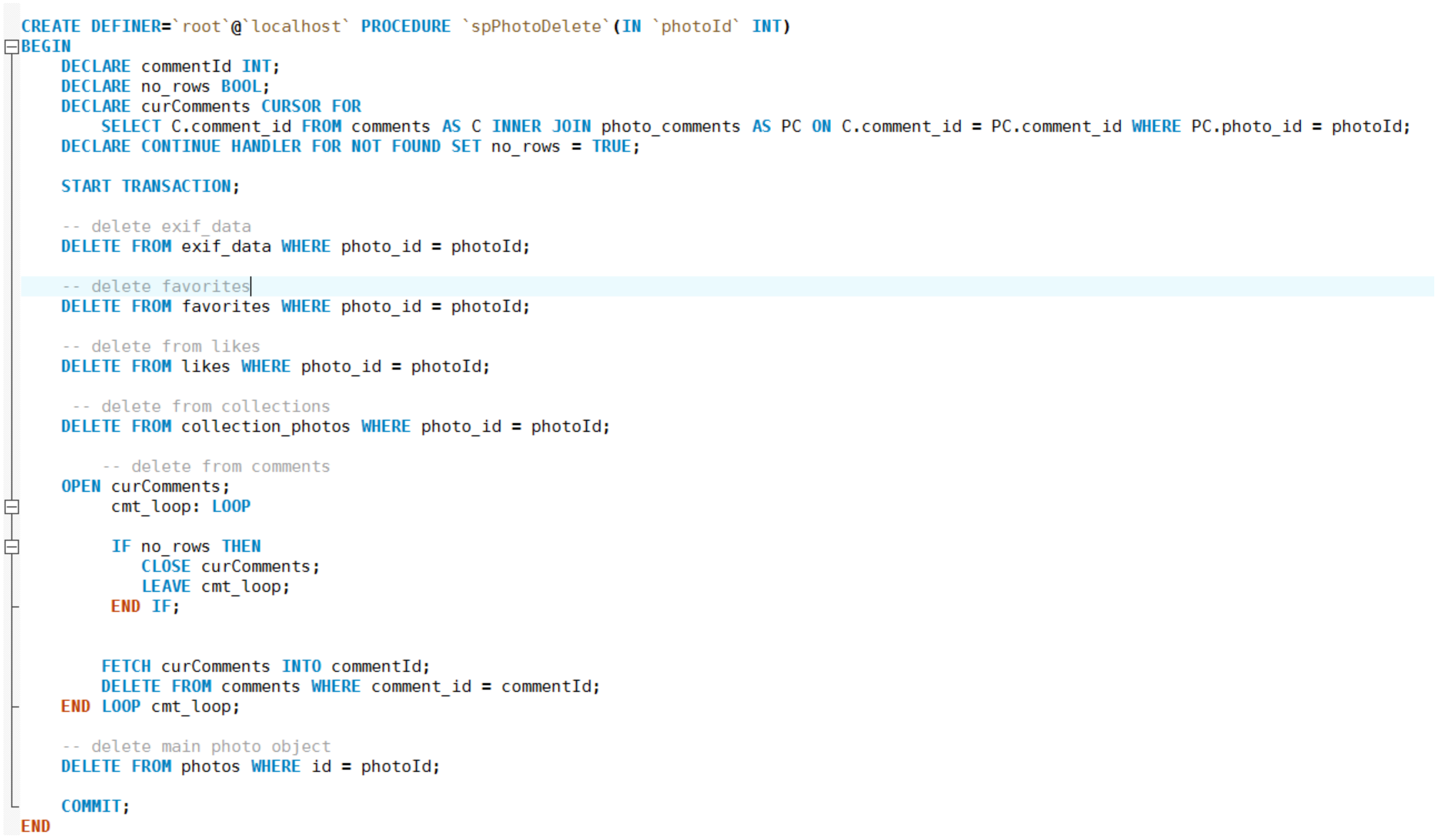
  Where timelapse(days) = (UploadTime – NOW)
- Executed using trigger after every comment/favorite or vote event on a photo.



```
-- UPDATE RATING FOR THE PHOTO

SELECT uploaded_date FROM photos WHERE id = NEW.photo_id INTO uploadDate;

SELECT TIMESTAMPDIFF(DAY,uploadDate,NOW()) INTO daysSinceUpload;

IF daysSinceUpload = 0 THEN
   SET daysSinceUpload = 1;
END IF;

UPDATE photos SET popularity = popularity + (1/daysSinceUpload)
WHERE id = NEW.photo_id;

-- END RATING
```

Stored Procedure

- The application uses one stored procedure : to delete photo object



```
CREATE DEFINER=`root`@`localhost` PROCEDURE `spPhotoDelete`(IN `photoId` INT)
BEGIN
    DECLARE commentId INT;
    DECLARE no_rows BOOL;
    DECLARE curComments CURSOR FOR
        SELECT C.comment_id FROM comments AS C INNER JOIN photo_comments AS PC ON C.comment_id = PC.comment_id WHERE PC.photo_id = photoId;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET no_rows = TRUE;

    START TRANSACTION;

    -- delete exif_data
    DELETE FROM exif_data WHERE photo_id = photoId;

    -- delete favorites
    DELETE FROM favorites WHERE photo_id = photoId;

    -- delete from likes
    DELETE FROM likes WHERE photo_id = photoId;

     -- delete from collections
    DELETE FROM collection_photos WHERE photo_id = photoId;

        -- delete from comments
    OPEN curComments;
         cmt_loop: LOOP

         IF no_rows THEN
            CLOSE curComments;
            LEAVE cmt_loop;
         END IF;


        FETCH curComments INTO commentId;
        DELETE FROM comments WHERE comment_id = commentId;
    END LOOP cmt_loop;

    -- delete main photo object
    DELETE FROM photos WHERE id = photoId;

    COMMIT;
END
```

Topics

- Introduction to PhotoShr
- Database Design
  - Entity Relationship Model
  - Triggers
    - Rating Algorithm
  - Stored Procedures
- Implementation
  - Tools and Technologies Used
  - Demo
- Future Enhancement

Tools and Technology Used

Tools and Technologies	Purpose
C#	Programming Language
ASP.NET MVC 3, CSS3, HTML5, JavaScript	Frontend
MySQL v5.6	RDBMS
Microsoft .NET Framework 4.5	Programming Framework
Twitter Bootstrap	CSS Framework
jQuery	JavaScript Library

Demo

Future Enhancements

- Future Enhancements
  - Better rating algorithm.
  - Image viewer with light-box functionality and slideshows.
  - Better search functionality.
  - Revenue generation model.

Thank you

Questions?

Source code

- Full source code available at github :
  - https://github.com/sumanshakya/PhotoShr/