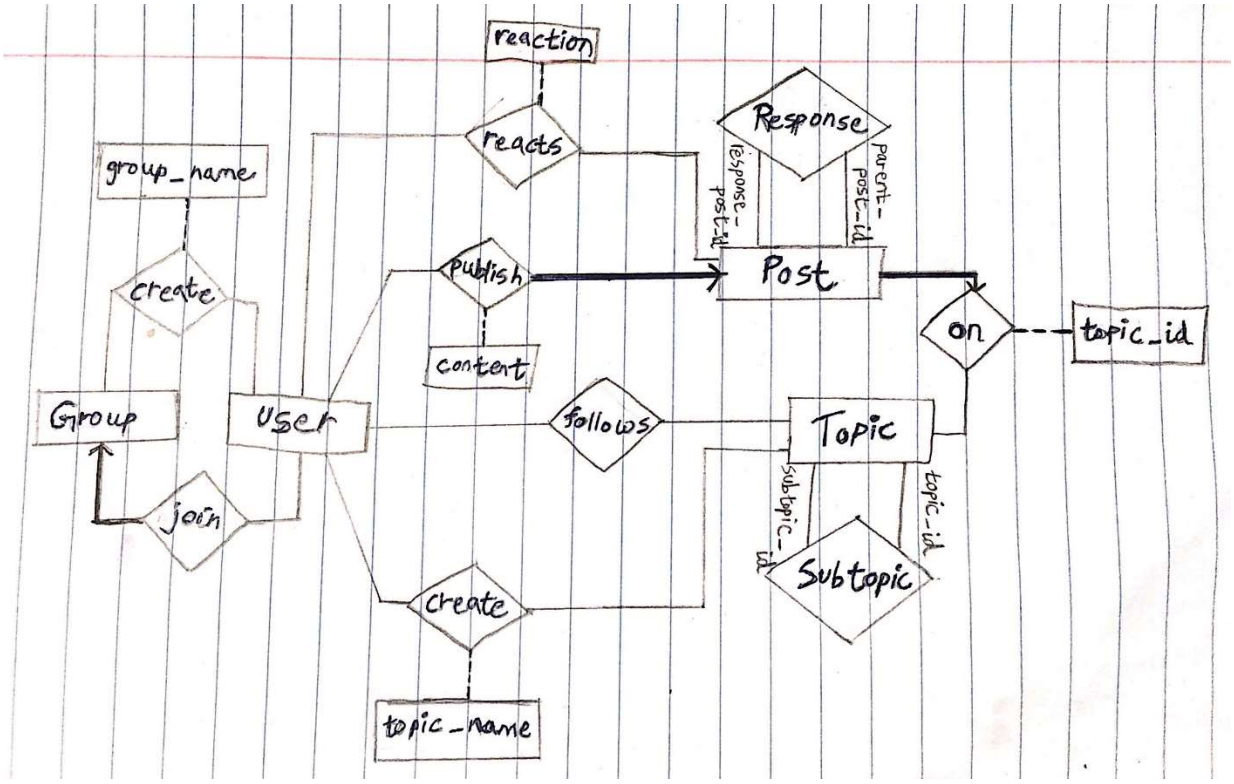
CHANDRAKIRAN'S SOCIAL NETWORK – ECE356 WINTER 2020

- SOCIAL_NETWORK SCHEMA

Welcome to Chadrakiran's social network! This social network is built using Node.js (implemented as Express for making client API's) and a MySQL database. It has basic functionalities such as topics, posts, groups, subscriptions, responses, and reactions (binary, like and dislike). The source data to populate the database is implemented in the SQL file itself, and not as an external file/entity. The schema is as follows:

- social_network
  - Tables
    - Following
    - Group_Member
    - Groups
    - Person
    - Post
    - Post_Reaction
    - Post_Response
    - Sub_Topic
    - Topic

- ER MODEL



- DESCRIPTION

In this social network, a user can:

- ***PUBLISH*** a post on a specific topic (given by topic_id). The descriptive attribute is *context*. This contains the content being published (text, image, etc.). The Post relationship is recursive to a response (parent_post_id and response_post_id). In other words, a response is recorded as a post characterized by a response_post_id to that parent_post_id. This logic allows users to respond to posts, as well as to responses themselves. Note the total participation of Post in the 'publish' and 'on' topic relations. This means that every post must be published by an existing user and every post must be on a specific topic/subtopic.

- ***FOLLOW*** a topic/subtopic of their liking (given by topic_id/subtopic_id). The Topic relation is recursive subtopics (topic_id and subtopic_id). In other words, a subtopic is characterized by a subtopic_id to a specific topic_id. Given these parameters, the logic allows any subtopic to a topic or even an existing subtopic, can be recorded.

- ***CREATE*** a topic. The descriptive attribute is topic_name. The user can, as mentioned above, publish a post on this newly created topic (given by topic_id)! Subtopics can also be created due to the recursive logic!

- ***CREATE*** a group. The descriptive attribute is group_name. There is no limit to the number of people in a group. The concept of 'sub-groups' is not being implemented here.

- ***JOIN*** a group. Note that the participation of Group here is complete. This means that a group must exist (given by group_id) in order to join it. It is not a weak entity relationship. This is because a group can have zero members!

- ***REACT*** to a post/response. The descriptive attribute is reaction. Reaction is being implemented as a Boolean value. So, the user can only like and dislike a post/response.

- ***VIEW*** their feed. This is not shown in the ER Model. Please refer to the command sheet.

- SUMMARY

Person (<u>id</u>, first_name, last_name, birthday, last_active, join_date)
Topic (<u>id</u>, topic_name)
Sub_Topic (topic_id, subtopic_id)
Following (person_id, topic_id)
Groups (<u>id</u>, group_name)
Group_Member (group_id, person_id)
Post (<u>id</u>, person_id, topic_id, content, post_created)
Post_Response (response_post_id, parent_post_id)
Post_Reaction (<u>post_id, person_id</u>, reaction)