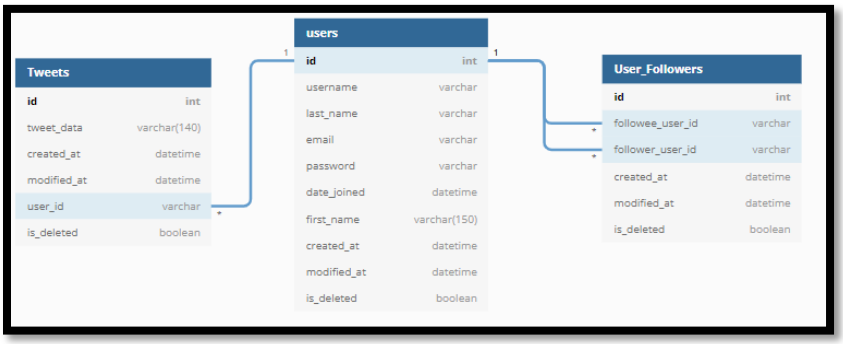
Assignment to Design Twitter System:

Twitter Database Schema:



API:

- **Unauthenticated user - Create user (Signup):**
  - **POST** signup\
    - Username:
    - Email address:
    - Password:
  - Validate the data and then add record in the Users table

- **Authenticated user:**
  - **Get** homepage\
    - Fetch all recent tweets made by the followees (people this user follows) of the logged in user
    - (**@user_id** = request.user.id)
    - (Select tweet_data from tweets where user_id in *(select followee_user_id where followee_user_id = **@user.id**)*
  - **POST** follow\{user_id}
    - Add entry in the User_Followers table with
    - followee_user_id = {user_id}
    - follower_user_id = **@user_id**
  - **POST** tweet\
    - (With tweet message in request body)
    - Add entry in the tweets table

Scalability and High-level considerations:

- Set up Load Balancer: Handle incoming requests better
- More reads expected than writes to the DB hence multiple read databases should be available
- Eventual consistency seems optimal choice as availability is important
- DB can be replicated (for better performance and fault tolerance)
- **Issues might arise when user has lot of tweets to be fetched – which are posted by follwees**
  - For improvements - we can setup another DB and keep adding tweets posted by the followees and store in a faster **NoSQL cache DB, use this DB to fetch timeline for the user**
  - This will reduce timeline load time as it would be present in the cache DB while using pre-processing approach