**Creating our Instagram like database**

**Step 1: What do we need to store**

* Photos
  + tags
* Captions and Hashtags
* Usernames
* Likes
* Comments
* Followers
* Following
* **Step 2 a:** Research – has anyone solved this problem before. Can you learn from their work and solutions to make your work faster and cleaner

**Step 2: Constraints**

* Each profile can only like a photo once
* People can follow someone and not be followed back and vice versa
* Username should be unique.
* Photos must be attached to a user
* Cant follow yourself
* Can only follow somebody one time

**Step 3: Table Design –** Brainstorm potential idea of what each table might consist of and identify some primary and foreign keys. Might not add all these features if too complex.

* Create tables for Users, Photos, Likes, Comments, and Followers.
* Each table needs a primary key for uniqueness.
* Use foreign keys to link tables for data consistency.

1. **Users Table**
   * User ID (Primary Key)
   * Username
   * Full Name
   * Email
   * Password (encrypted)
   * Profile Picture
   * Bio
   * Registration Date
   * Other User-Related Information
2. **Photos Table**
   * Photo ID (Primary Key)
   * User ID (Foreign Key to Users Table)
   * Photo URL
   * Caption
   * Timestamp
   * Location
   * Tags
   * Other Photo-Related Information
3. **Likes Table**
   * Like ID (Primary Key)
   * User ID (Foreign Key to Users Table)
   * Photo ID (Foreign Key to Photos Table)
   * Timestamp
   * (To enforce the constraint, each row in this table represents a user liking a photo.)
4. **Comments Table**
   * Comment ID (Primary Key)
   * User ID (Foreign Key to Users Table)
   * Photo ID (Foreign Key to Photos Table)
   * Comment Text
   * Timestamp
5. **Followers Table**
   * Follower ID (Primary Key)
   * Follower User ID (Foreign Key to Users Table)
   * Followed User ID (Foreign Key to Users Table)
   * Timestamp
   * (This table allows users to follow other users. No need for reciprocal entries as not all follows are mutual.)
6. Hashtags Table
   * To speed things up we will create 2 tables to handle hashtags this article suggests it is fastest

http://howto.philippkeller.com/2005/06/19/Tagsystems-performance-tests/

* + Doing this we reduce duplication
  + We cam store more info about the tags like time..
  + Photo\_tags
    1. Photo\_id
    2. Tag\_id
  + Tags
    1. Id
    2. Tag\_name

**Step 5:** Sketch out database table schemas to make things easier to look at:

A diagram of a data base

Description automatically generated

**Step 6: Data Types and Attributes**

* Decide on data types for attributes (e.g., strings for names and email, date/time for timestamps).
* Opt for the right data types to save space and enhance performance.

**Step 7: Refining and Optimizing**

* Remember, it's an iterative process.
* Normalize the database to minimize redundancy.
* Optimize queries for faster results.
* Plan indexing strategies for quicker data retrieval.

**Step 8: Testing and Scaling**

* Testing is crucial for data consistency and accuracy.
* Simulate real-world scenarios for issue spotting.
* As the app grows, think about scaling techniques like partitioning