Database Schema

```
mysql> CREATE DATABASE IF NOT EXISTS SocialMediaPlatform;
Query OK, 1 row affected (0.01 sec)

mysql> USE SocialMediaPlatform;
Database changed
mysql>
mysql> CREATE TABLE IF NOT EXISTS Users (
    ->     user_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     username VARCHAR(50) NOT NULL UNIQUE,
    ->     email VARCHAR(255) NOT NULL UNIQUE,
    ->     password_hash VARCHAR(255) NOT NULL,
    ->     date_of_birth DATE,
    ->     profile_picture_url VARCHAR(255),
    ->     bio TEXT,
    ->     join_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     post_count INT DEFAULT 0,
    ->     follower_count INT DEFAULT 0,
    ->     following_count INT DEFAULT 0
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql>
mysql> CREATE TABLE IF NOT EXISTS Posts (
    ->     post_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     user_id INT NOT NULL,
    ->     post_text TEXT,
    ->     media_url VARCHAR(255),
    ->     post_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     likes_count INT DEFAULT 0,
    ->     comments_count INT DEFAULT 0,
    ->     FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE ON UPDATE CASCADE
    -> );
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> CREATE TABLE IF NOT EXISTS Comments (
    ->     comment_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     post_id INT NOT NULL,
    ->     user_id INT NOT NULL,
    ->     comment_text TEXT NOT NULL,
    ->     comment_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     FOREIGN KEY (post_id) REFERENCES Posts(post_id) ON DELETE CASCADE ON UPDATE CASCADE,
    ->     FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE ON UPDATE CASCADE
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql>
mysql> CREATE TABLE IF NOT EXISTS Likes (
    ->     like_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     post_id INT NOT NULL,
    ->     user_id INT NOT NULL,
    ->     like_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     FOREIGN KEY (post_id) REFERENCES Posts(post_id) ON DELETE CASCADE ON UPDATE CASCADE,
    ->     FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE ON UPDATE CASCADE,
    ->     UNIQUE KEY unique_like (post_id, user_id)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql>
mysql> CREATE TABLE IF NOT EXISTS Follows (
    ->     follow_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     follower_id INT NOT NULL,
    ->     following_id INT NOT NULL,
    ->     follow_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     FOREIGN KEY (follower_id) REFERENCES Users(user_id) ON DELETE CASCADE ON UPDATE CASCADE,
    ->     FOREIGN KEY (following_id) REFERENCES Users(user_id) ON DELETE CASCADE ON UPDATE CASCADE,
    ->     UNIQUE KEY unique_follow (follower_id, following_id)
    -> );
Query OK, 0 rows affected (0.03 sec)
```

```
mysql>
mysql> CREATE TABLE IF NOT EXISTS Messages (
    ->     message_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     sender_id INT NOT NULL,
    ->     receiver_id INT NOT NULL,
    ->     message_text TEXT NOT NULL,
    ->     message_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     is_read BOOLEAN DEFAULT FALSE,
    ->     FOREIGN KEY (sender_id) REFERENCES Users(user_id) ON DELETE CASCADE ON UPDATE CASCADE,
    ->     FOREIGN KEY (receiver_id) REFERENCES Users(user_id) ON DELETE CASCADE ON UPDATE CASCADE
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql>
mysql> CREATE TABLE IF NOT EXISTS Notifications (
    ->     notification_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     user_id INT NOT NULL,
    ->     notifier_id INT,
    ->     notification_type ENUM('new_like', 'new_comment', 'new_follower', 'new_message', 'post_mention') NOT NULL,
    ->     source_id INT,
    ->     notification_text VARCHAR(255),
    ->     notification_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     is_read BOOLEAN DEFAULT FALSE,
    ->     FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE ON UPDATE CASCADE,
    ->     FOREIGN KEY (notifier_id) REFERENCES Users(user_id) ON DELETE SET NULL ON UPDATE CASCADE
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql>
```

Insert Sample Data

```
mysql>
mysql> INSERT INTO Posts (user_id, post_text, media_url) VALUES
    -> (1, 'Just enjoyed a beautiful sunset! #nature #travel', 'http://example.com/media/sunset1.jpg'),
    -> (2, 'Working on a new project. Stay tuned! #coding #development', NULL),
    -> (1, 'My thoughts on the latest tech trends. #technology', NULL),
    -> (3, 'Found this cool comic strip today!', 'http://example.com/media/comic1.png'),
    -> (4, 'Remember to be kind to one another. #inspiration', NULL),
    -> (1, 'Delicious brunch this morning! #foodie', 'http://example.com/media/brunch.jpg');
Query OK, 6 rows affected (0.00 sec)
Records: 6  Duplicates: 0  Warnings: 0

mysql>
mysql> INSERT INTO Comments (post_id, user_id, comment_text) VALUES
    -> (1, 2, 'Wow, that looks amazing Alice!'),
    -> (1, 3, 'Stunning shot!'),
    -> (2, 1, 'Can''t wait to see it, Bob!'),
    -> (2, 4, 'Sounds exciting! Good luck!'),
    -> (4, 1, 'Love this message, Diana.'),
    -> (5, 2, 'So true!');
Query OK, 6 rows affected (0.00 sec)
Records: 6  Duplicates: 0  Warnings: 0

mysql>
mysql> INSERT INTO Likes (post_id, user_id) VALUES
    -> (1, 2), (1, 3), (1, 4),
    -> (2, 1), (2, 3),
    -> (3, 2),
    -> (4, 1), (4, 2), (4, 3), (4, 5),
    -> (5, 1), (5, 3);
Query OK, 12 rows affected (0.00 sec)
Records: 12  Duplicates: 0  Warnings: 0
```

```
mysql>
mysql> INSERT INTO Follows (follower_id, following_id) VALUES (1, 2), (1, 4);
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> INSERT INTO Follows (follower_id, following_id) VALUES (2, 1);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO Follows (follower_id, following_id) VALUES (3, 1), (3, 2);
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> INSERT INTO Follows (follower_id, following_id) VALUES (4, 1);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO Follows (follower_id, following_id) VALUES (5,1), (5,2), (5,3), (5,4);
Query OK, 4 rows affected (0.00 sec)
Records: 4  Duplicates: 0  Warnings: 0
```

```
mysql> INSERT INTO Messages (sender_id, receiver_id, message_text, is_read) VALUES
    -> (1, 2, 'Hey Bob, how is the project going?', FALSE),
    -> (2, 1, 'Hi Alice! It''s going well, almost done with the prototype.', TRUE),
    -> (1, 2, 'Great to hear! Let me know if you need any help testing.', FALSE),
    -> (3, 4, 'Hi Diana, loved your recent post!', FALSE),
    -> (4, 3, 'Thanks Charlie! Appreciate it.', TRUE);
Query OK, 5 rows affected (0.00 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql>
mysql> INSERT INTO Notifications (user_id, notifier_id, notification_type, source_id, notification_text, is_read) VALUES

    -> (1, 2, 'new_like', 1, 'Bob Builder liked your post.', FALSE),
    -> (1, 2, 'new_comment', 1, 'Bob Builder commented on your post: "Wow, that looks amazing Alice!"', FALSE),
    -> (2, 1, 'new_comment', 2, 'Alice Wonder commented on your post: "Can''t wait to see it, Bob!"', FALSE),
    -> (1, 2, 'new_follower', 2, 'Bob Builder started following you.', FALSE),
    -> (2, 1, 'new_message', 1, 'You have a new message from Alice Wonder.', FALSE);
Query OK, 5 rows affected (0.00 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql>
mysql> UPDATE Users u SET post_count = (SELECT COUNT(*) FROM Posts p WHERE p.user_id = u.user_id);
Query OK, 4 rows affected (0.00 sec)
Rows matched: 5  Changed: 4  Warnings: 0

mysql> UPDATE Users u SET follower_count = (SELECT COUNT(*) FROM Follows f WHERE f.following_id = u.user_id);
Query OK, 4 rows affected (0.00 sec)
Rows matched: 5  Changed: 4  Warnings: 0

mysql> UPDATE Users u SET following_count = (SELECT COUNT(*) FROM Follows f WHERE f.follower_id = u.user_id);
Query OK, 5 rows affected (0.00 sec)
Rows matched: 5  Changed: 5  Warnings: 0

mysql> UPDATE Posts p SET likes_count = (SELECT COUNT(*) FROM Likes l WHERE l.post_id = p.post_id);
Query OK, 5 rows affected (0.00 sec)
Rows matched: 6  Changed: 5  Warnings: 0

mysql> UPDATE Posts p SET comments_count = (SELECT COUNT(*) FROM Comments c WHERE c.post_id = p.post_id);
Query OK, 4 rows affected (0.01 sec)
Rows matched: 6  Changed: 4  Warnings: 0
```

Queries

- Retrieve the posts and activities of a user's timeline.

```
mysql> SELECT
    ->     p.post_id,
    ->     p.post_text,
    ->     p.media_url,
    ->     p.post_date,
    ->     u.user_id AS author_id,
    ->     u.username AS author_username,
    ->     u.profile_picture_url AS author_profile_picture,
    ->     p.likes_count,
    ->     p.comments_count
    -> FROM Posts p
    -> JOIN Users u ON p.user_id = u.user_id
    -> JOIN Follows f ON p.user_id = f.following_id
    -> WHERE f.follower_id = 1
    -> UNION ALL
    -> SELECT
    ->     p.post_id,
    ->     p.post_text,
    ->     p.media_url,
    ->     p.post_date,
    ->     u.user_id AS author_id,
    ->     u.username AS author_username,
    ->     u.profile_picture_url AS author_profile_picture,
    ->     p.likes_count,
    ->     p.comments_count
    -> FROM Posts p
    -> JOIN Users u ON p.user_id = u.user_id
    -> WHERE p.user_id = 1
    -> ORDER BY post_date DESC
    -> LIMIT 20;
```

```
    -> LIMIT 20;
+---------+-----------------------------------------------------------------+-----------------------------------+--------------------+-----------+----------
-------+-----------------------------------+-------------+----------------+
| post_id | post_text                                                       | media_url                         | post_date          | author_id | author_us
ername | author_profile_picture            | likes_count | comments_count |
+---------+-----------------------------------------------------------------+-----------------------------------+--------------------+-----------+----------
-------+-----------------------------------+-------------+----------------+
|       2 | Working on a new project. Stay tuned! #coding #development | NULL                              | 2025-05-07 23:19:34 |         2 | bob_build
er     | http://example.com/pics/bob.jpg   |           2 |              2 |
|       5 | Remember to be kind to one another. #inspiration           | NULL                              | 2025-05-07 23:19:34 |         4 | diana_pri
nce    | http://example.com/pics/diana.jpg |           2 |              1 |
|       1 | Just enjoyed a beautiful sunset! #nature #travel           | http://example.com/media/sunset1.jpg | 2025-05-07 23:19:34 |         1 | alice_won
der    | http://example.com/pics/alice.jpg |           3 |              2 |
|       3 | My thoughts on the latest tech trends. #technology         | NULL                              | 2025-05-07 23:19:34 |         1 | alice_won
der    | http://example.com/pics/alice.jpg |           1 |              0 |
|       6 | Delicious brunch this morning! #foodie                     | http://example.com/media/brunch.jpg  | 2025-05-07 23:19:34 |         1 | alice_won
der    | http://example.com/pics/alice.jpg |           0 |              0 |
+---------+-----------------------------------------------------------------+-----------------------------------+--------------------+-----------+----------
-------+-----------------------------------+-------------+----------------+
5 rows in set (0.01 sec)
```

- Retrieve the comments and likes for a specific post.

```
mysql> SELECT
    ->     c.comment_id,
    ->     c.comment_text,
    ->     c.comment_date,
    ->     u.user_id AS commenter_id,
    ->     u.username AS commenter_username,
    ->     u.profile_picture_url AS commenter_profile_picture
    -> FROM Comments c
    -> JOIN Users u ON c.user_id = u.user_id
    -> WHERE c.post_id = 1
    -> ORDER BY c.comment_date ASC;
+------------+-----------------------------+---------------------+--------------+--------------------+-----------------------------------+
| comment_id | comment_text                | comment_date        | commenter_id | commenter_username | commenter_profile_picture         |
+------------+-----------------------------+---------------------+--------------+--------------------+-----------------------------------+
|          1 | Wow, that looks amazing Alice! | 2025-05-07 23:19:34 |            2 | bob_builder        | http://example.com/pics/bob.jpg |
|          2 | Stunning shot!              | 2025-05-07 23:19:34 |            3 | charlie_brown      | NULL                              |
+------------+-----------------------------+---------------------+--------------+--------------------+-----------------------------------+
2 rows in set (0.00 sec)
```

```
mysql>
mysql> SELECT
    ->     l.like_id,
    ->     l.like_date,
    ->     u.user_id AS liker_id,
    ->     u.username AS liker_username,
    ->     u.profile_picture_url AS liker_profile_picture
    -> FROM Likes l
    -> JOIN Users u ON l.user_id = u.user_id
    -> WHERE l.post_id = 1
    -> ORDER BY l.like_date DESC;
+---------+---------------------+----------+----------------+-----------------------------------+
| like_id | like_date           | liker_id | liker_username | liker_profile_picture             |
+---------+---------------------+----------+----------------+-----------------------------------+
|       1 | 2025-05-07 23:19:34 |        2 | bob_builder    | http://example.com/pics/bob.jpg   |
|       2 | 2025-05-07 23:19:34 |        3 | charlie_brown  | NULL                              |
|       3 | 2025-05-07 23:19:34 |        4 | diana_prince   | http://example.com/pics/diana.jpg |
+---------+---------------------+----------+----------------+-----------------------------------+
3 rows in set (0.00 sec)

mysql>
```

- Retrieve the list of followers for a user.

```
mysql> SELECT
    ->     u.user_id AS follower_id,
    ->     u.username AS follower_username,
    ->     u.profile_picture_url AS follower_profile_picture,
    ->     f.follow_date
    -> FROM Users u
    -> JOIN Follows f ON u.user_id = f.follower_id
    -> WHERE f.following_id = 1
    -> ORDER BY f.follow_date DESC;
+-------------+-------------------+-----------------------------------+---------------------+
| follower_id | follower_username | follower_profile_picture          | follow_date         |
+-------------+-------------------+-----------------------------------+---------------------+
|           2 | bob_builder       | http://example.com/pics/bob.jpg   | 2025-05-07 23:19:34 |
|           3 | charlie_brown     | NULL                              | 2025-05-07 23:19:34 |
|           4 | diana_prince      | http://example.com/pics/diana.jpg | 2025-05-07 23:19:34 |
|           5 | edward_scissor    | http://example.com/pics/edward.jpg | 2025-05-07 23:19:34 |
+-------------+-------------------+-----------------------------------+---------------------+
4 rows in set (0.00 sec)
```

- Retrieve unread messages for a user.

```
mysql> SELECT
    ->     m.message_id,
    ->     m.message_text,
    ->     m.message_date,
    ->     s.user_id AS sender_id,
    ->     s.username AS sender_username,
    ->     s.profile_picture_url AS sender_profile_picture
    -> FROM Messages m
    -> JOIN Users s ON m.sender_id = s.user_id
    -> WHERE m.receiver_id = 1 AND m.is_read = FALSE
    -> ORDER BY m.message_date DESC;
Empty set (0.00 sec)

mysql>
```

- Retrieve the most liked posts.

```
mysql> SELECT
    ->     p.post_id,
    ->     p.post_text,
    ->     p.media_url,
    ->     u.username AS author_username,
    ->     p.likes_count,
    ->     p.comments_count,
    ->     p.post_date
    -> FROM Posts p
    -> JOIN Users u ON p.user_id = u.user_id
    -> ORDER BY p.likes_count DESC
    -> LIMIT 10;
+---------+-------------------------------------------------------------+-----------------------------------+-----------------+-------------+-------------
----+---------------------+
| post_id | post_text                                                   | media_url                         | author_username | likes_count | comments_co
unt | post_date           |
+---------+-------------------------------------------------------------+-----------------------------------+-----------------+-------------+-------------
----+---------------------+
|       4 | Found this cool comic strip today!                          | http://example.com/media/comic1.png | charlie_brown   |           4 |
  1 | 2025-05-07 23:19:34 |
|       1 | Just enjoyed a beautiful sunset! #nature #travel            | http://example.com/media/sunset1.jpg | alice_wonder    |           3 |
  2 | 2025-05-07 23:19:34 |
|       2 | Working on a new project. Stay tuned! #coding #development | NULL                              | bob_builder     |           2 |
  2 | 2025-05-07 23:19:34 |
|       5 | Remember to be kind to one another. #inspiration            | NULL                              | diana_prince    |           2 |
  1 | 2025-05-07 23:19:34 |
|       3 | My thoughts on the latest tech trends. #technology          | NULL                              | alice_wonder    |           1 |
  0 | 2025-05-07 23:19:34 |
|       6 | Delicious brunch this morning! #foodie                      | http://example.com/media/brunch.jpg | alice_wonder    |           0 |
  0 | 2025-05-07 23:19:34 |
+---------+-------------------------------------------------------------+-----------------------------------+-----------------+-------------+-------------
----+---------------------+
6 rows in set (0.00 sec)
```

- Retrieve the latest notifications for a user.

```
mysql> SELECT
    ->     notification_id,
    ->     notification_type,
    ->     notification_text,
    ->     source_id,
    ->     notifier_id,
    ->     (SELECT username FROM Users WHERE user_id = notifier_id) as notifier_username,
    ->     notification_date,
    ->     is_read
    -> FROM Notifications
    -> WHERE user_id = 1
    -> ORDER BY notification_date DESC
    -> LIMIT 10;
+-----------------+-------------------+-----------------------------------------------------------------------+-----------+-------------+--------------------
+---------------------+---------+
| notification_id | notification_type | notification_text                                                     | source_id | notifier_id | notifier_username
| notification_date   | is_read |
+-----------------+-------------------+-----------------------------------------------------------------------+-----------+-------------+--------------------
+---------------------+---------+
|               1 | new_like          | Bob Builder liked your post.                                          |         1 |           2 | bob_builder
| 2025-05-07 23:19:34 |       0 |
|               2 | new_comment       | Bob Builder commented on your post: "Wow, that looks amazing Alice!" |         1 |           2 | bob_builder
| 2025-05-07 23:19:34 |       0 |
|               4 | new_follower      | Bob Builder started following you.                                    |         2 |           2 | bob_builder
| 2025-05-07 23:19:34 |       0 |
+-----------------+-------------------+-----------------------------------------------------------------------+-----------+-------------+--------------------
+---------------------+---------+
3 rows in set (0.00 sec)
```

Data Modification

- Add a new post to the platform.

```
mysql> INSERT INTO Posts (user_id, post_text, media_url)
    -> VALUES (2, 'Just launched my new website! Check it out. #webdev #launch', 'http://example.com/media/website_launch.png');
Query OK, 1 row affected (0.01 sec)

mysql>
```

- Comment on a post.

```
mysql> INSERT INTO Comments (post_id, user_id, comment_text)
    -> VALUES (2, 1, 'Looks fantastic, Bob! Congratulations on the launch!');
Query OK, 1 row affected (0.01 sec)

mysql>
```

- Update user profile information.

```
mysql> UPDATE Users
    -> SET
    ->     bio = 'Traveling the globe and sharing stories. Love photography!',
    ->     profile_picture_url = 'http://example.com/pics/alice_new.jpg'
    -> WHERE user_id = 1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
```

- Remove a like from a post.

```
mysql> DELETE FROM Likes
    -> WHERE post_id = 1 AND user_id = 3;
Query OK, 1 row affected (0.01 sec)

mysql>
```

Complex Queries

- Identify users with the most followers.

```
mysql> SELECT
    ->     u.user_id,
    ->     u.username,
    ->     u.profile_picture_url,
    ->     u.follower_count
    -> FROM Users u
    -> ORDER BY u.follower_count DESC
    -> LIMIT 10;
+---------+----------------+-------------------------------------+----------------+
| user_id | username       | profile_picture_url                 | follower_count |
+---------+----------------+-------------------------------------+----------------+
|       1 | alice_wonder   | http://example.com/pics/alice_new.jpg |              4 |
|       2 | bob_builder    | http://example.com/pics/bob.jpg       |              3 |
|       4 | diana_prince   | http://example.com/pics/diana.jpg     |              2 |
|       3 | charlie_brown  | NULL                                |              1 |
|       5 | edward_scissor | http://example.com/pics/edward.jpg    |              0 |
+---------+----------------+-------------------------------------+----------------+
5 rows in set (0.00 sec)

mysql>
```

- Find the most active users based on post count and interaction.

```
mysql> SELECT
    ->     u.user_id,
    ->     u.username,
    ->     u.post_count,
    ->     SUM(p.likes_count) AS total_likes_received,
    ->     SUM(p.comments_count) AS total_comments_received,
    ->     (u.post_count + SUM(p.likes_count) + SUM(p.comments_count)) AS activity_score
    -> FROM Users u
    -> LEFT JOIN Posts p ON u.user_id = p.user_id
    -> GROUP BY u.user_id, u.username, u.post_count
    -> ORDER BY activity_score DESC
    -> LIMIT 10;
+---------+---------------+------------+---------------------+-------------------------+----------------+
| user_id | username      | post_count | total_likes_received | total_comments_received | activity_score |
+---------+---------------+------------+---------------------+-------------------------+----------------+
|       1 | alice_wonder  |          3 |                   4 |                       2 |              9 |
|       3 | charlie_brown |          1 |                   4 |                       1 |              6 |
|       2 | bob_builder   |          1 |                   2 |                       2 |              5 |
|       4 | diana_prince  |          1 |                   2 |                       1 |              4 |
|       5 | edward_scissor |          0 |                NULL |                    NULL |           NULL |
+---------+---------------+------------+---------------------+-------------------------+----------------+
5 rows in set (0.00 sec)
```

- Calculate the average number of comments per post.

```
mysql>
mysql> SELECT AVG(num_comments) AS average_comments_per_post
    -> FROM (
    ->     SELECT p.post_id, COUNT(c.comment_id) AS num_comments
    ->     FROM Posts p
    ->     LEFT JOIN Comments c ON p.post_id = c.post_id
    ->     GROUP BY p.post_id
    -> ) AS post_comment_counts;
+---------------------------+
| average_comments_per_post |
+---------------------------+
|                    1.0000 |
+---------------------------+
1 row in set (0.00 sec)
```

Advanced Topics

- Automatically notify users of new messages.

```
mysql> DELIMITER //
mysql> CREATE TRIGGER AfterPostInsert
    -> AFTER INSERT ON Posts
    -> FOR EACH ROW
    -> BEGIN
    ->     UPDATE Users SET post_count = post_count + 1 WHERE user_id = NEW.user_id;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql>
mysql> DELIMITER //
mysql> CREATE TRIGGER AfterPostDelete
    -> AFTER DELETE ON Posts
    -> FOR EACH ROW
    -> BEGIN
    ->     UPDATE Users SET post_count = post_count - 1 WHERE user_id = OLD.user_id;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql>
mysql> DELIMITER //
mysql> CREATE TRIGGER AfterLikeInsert
    -> AFTER INSERT ON Likes
    -> FOR EACH ROW
    -> BEGIN
    ->     UPDATE Posts SET likes_count = likes_count + 1 WHERE post_id = NEW.post_id;
    -> END //
Query OK, 0 rows affected (0.00 sec)

mysql> DELIMITER ;
mysql>
mysql> DELIMITER //
mysql> CREATE TRIGGER AfterLikeDelete
    -> AFTER DELETE ON Likes
    -> FOR EACH ROW
    -> BEGIN
    ->     UPDATE Posts SET likes_count = likes_count - 1 WHERE post_id = OLD.post_id;
    -> END //
```

```
mysql> DELIMITER //
mysql> CREATE TRIGGER AfterCommentInsert
    -> AFTER INSERT ON Comments
    -> FOR EACH ROW
    -> BEGIN
    ->     UPDATE Posts SET comments_count = comments_count + 1 WHERE post_id = NEW.post_id;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql>
mysql> DELIMITER //
mysql> CREATE TRIGGER AfterCommentDelete
    -> AFTER DELETE ON Comments
    -> FOR EACH ROW
    -> BEGIN
    ->     UPDATE Posts SET comments_count = comments_count - 1 WHERE post_id = OLD.post_id;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql>
mysql> DELIMITER //
mysql> CREATE TRIGGER AfterFollowInsert
    -> AFTER INSERT ON Follows
    -> FOR EACH ROW
    -> BEGIN
    ->     UPDATE Users SET following_count = following_count + 1 WHERE user_id = NEW.follower_id;
    ->     UPDATE Users SET follower_count = follower_count + 1 WHERE user_id = NEW.following_id;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
```

```
mysql>
mysql> DELIMITER //
mysql> CREATE TRIGGER AfterFollowDelete
    -> AFTER DELETE ON Follows
    -> FOR EACH ROW
    -> BEGIN
    ->     UPDATE Users SET following_count = following_count - 1 WHERE user_id = OLD.follower_id;
    ->     UPDATE Users SET follower_count = follower_count - 1 WHERE user_id = OLD.following_id;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
```

- Update post counts and follower counts for users.

```
mysql> DELIMITER //
mysql> CREATE TRIGGER NotifyOnNewLike
    -> AFTER INSERT ON Likes
    -> FOR EACH ROW
    -> BEGIN
    ->     DECLARE post_author_id INT;
    ->     DECLARE notifier_username VARCHAR(50);
    ->     SELECT user_id INTO post_author_id FROM Posts WHERE post_id = NEW.post_id;
    ->     SELECT username INTO notifier_username FROM Users WHERE user_id = NEW.user_id;
    ->
    ->     IF post_author_id <> NEW.user_id THEN
    ->         INSERT INTO Notifications (user_id, notifier_id, notification_type, source_id, notification_text)
    ->         VALUES (post_author_id, NEW.user_id, 'new_like', NEW.post_id, CONCAT(notifier_username, ' liked your post.'));
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql>
mysql> DELIMITER //
mysql> CREATE TRIGGER NotifyOnNewComment
    -> AFTER INSERT ON Comments
    -> FOR EACH ROW
    -> BEGIN
    ->     DECLARE post_author_id INT;
    ->     DECLARE commenter_username VARCHAR(50);
    ->     SELECT user_id INTO post_author_id FROM Posts WHERE post_id = NEW.post_id;
    ->     SELECT username INTO commenter_username FROM Users WHERE user_id = NEW.user_id;
    ->
    ->     IF post_author_id <> NEW.user_id THEN
    ->         INSERT INTO Notifications (user_id, notifier_id, notification_type, source_id, notification_text)
    ->         VALUES (post_author_id, NEW.user_id, 'new_comment', NEW.post_id, CONCAT(commenter_username, ' commented on your post: "', LEFT(NEW.comment_t
ext, 50), IF(LENGTH(NEW.comment_text) > 50, '...', '"')));
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> DELIMITER ;
mysql>
mysql> DELIMITER //
mysql> CREATE TRIGGER NotifyOnNewFollower
    -> AFTER INSERT ON Follows
    -> FOR EACH ROW
    -> BEGIN
    ->     DECLARE follower_username VARCHAR(50);
    ->     SELECT username INTO follower_username FROM Users WHERE user_id = NEW.follower_id;
    ->
    ->     INSERT INTO Notifications (user_id, notifier_id, notification_type, source_id, notification_text)
    ->     VALUES (NEW.following_id, NEW.follower_id, 'new_follower', NEW.follower_id, CONCAT(follower_username, ' started following you.'));
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql>
mysql> DELIMITER //
mysql> CREATE TRIGGER NotifyOnNewMessage
    -> AFTER INSERT ON Messages
    -> FOR EACH ROW
    -> BEGIN
    ->     DECLARE sender_username VARCHAR(50);
    ->     SELECT username INTO sender_username FROM Users WHERE user_id = NEW.sender_id;
    ->
    ->     INSERT INTO Notifications (user_id, notifier_id, notification_type, source_id, notification_text)
    ->     VALUES (NEW.receiver_id, NEW.sender_id, 'new_message', NEW.message_id, CONCAT('You have a new message from ', sender_username, '.'));
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql>
```

- Generate personalized recommendations for users to follow.

```
mysql>
mysql> DELIMITER //
mysql> CREATE PROCEDURE GetFollowRecommendations(IN current_user_id INT, IN limit_count INT)
    -> BEGIN
    ->     SELECT
    ->         u_to_follow.user_id,
    ->         u_to_follow.username,
    ->         u_to_follow.profile_picture_url,
    ->         u_to_follow.bio,
    ->         COUNT(DISTINCT f1.follower_id) as mutual_follows_count
    ->     FROM Users u_to_follow
    ->     JOIN Follows f2 ON u_to_follow.user_id = f2.following_id
    ->     JOIN Follows f1 ON f2.follower_id = f1.following_id AND f1.follower_id = current_user_id
    ->     WHERE u_to_follow.user_id <> current_user_id
    ->     AND u_to_follow.user_id NOT IN (SELECT following_id FROM Follows WHERE follower_id = current_user_id)
    ->     GROUP BY u_to_follow.user_id, u_to_follow.username, u_to_follow.profile_picture_url, u_to_follow.bio
    ->     ORDER BY mutual_follows_count DESC, RAND()
    ->     LIMIT limit_count;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql>
```