```
CREATE SCHEMA SocialDB DEFAULT CHARACTER SET 'UTF8MB4';
-- UTF8MBR IS USE FOR ASCII (American Standard Code for Information Interchange)
USE SocialDB
CREATE TABLE UserProfile (
UserProfileID INTEGER PRIMARY KEY AUTO_INCREMENT,
UserName      VARCHAR(20) UNIQUE NOT NULL,
Password      VARCHAR(20) NOT NULL,
Email         VARCHAR(40) UNIQUE NOT NULL,
FirstName     VARCHAR(20) NOT NULL,
MiddleName    VARCHAR(20),
LastName      VARCHAR(20) NOT NULL,
CreatedDTTM   DATETIME NOT NULL DEFAULT NOW(),
UpdatedDTTM   DATETIME,
IsDeleted     BIT NOT NULL DEFAULT 0
);

INSERT INTO UserProfile(UserName, Password, Email, FirstName, LastName)
VALUES('a','a','a@test.com','a','a');

INSERT INTO UserProfile(UserName, Password, Email, FirstName, LastName)
VALUES('b','b','b@test.com','b','b');

INSERT INTO UserProfile(UserName, Password, Email, FirstName, LastName)
VALUES('c','c','c@test.com','c','c');

INSERT INTO UserProfile(UserName, Password, Email, FirstName, LastName)
VALUES('d','d','d@test.com','d','d');

SELECT * FROM UserProfile;

-- UserProfileExt Table <--1-1--> UserProfile
-- tinytext = 255, text = 64kb, mediumtext = 16mb, longtext = 4gb (memory size for each category)

CREATE TABLE UserProfileExt (
UserProfileExtID  INTEGER PRIMARY KEY AUTO_INCREMENT,
UserProfileID     INTEGER UNIQUE NOT NULL,
ProfileImage      VARCHAR(100),
Phone             VARCHAR(12),
Website           VARCHAR(100),
HeadLine          VARCHAR(256),
Country           VARCHAR(50),
Summary           text,
CONSTRAINT fk_userprofileid FOREIGN KEY(UserProfileID) REFERENCES UserProfile(UserProfileID)
);

INSERT INTO UserProfileExt(UserProfileID, ProfileImage, Phone)
VALUES(1, '/storage/1/image.png', '1234');

SELECT * FROM UserProfileExt;

-- To Displey user info along with user profile extentions.

SELECT * FROM UserProfile AS u
LEFT OUTER JOIN UserProfileExt AS upe ON (u.UserProfileID = upe.UserProfileID);

-- User Connections Table
-- Many users can get connected with many users.
-- Many users can follow many users.

CREATE TABLE UserConnections(
UserOne      INTEGER NOT NULL,
UserTwo      INTEGER NOT NULL,
ISConnection  BIT NOT NULL,
ISFollower    BIT NOT NULL,
ConnectedDTTM DATETIME NOT NULL DEFAULT NOW(),
CONSTRAINT fk_userone_userprofid FOREIGN KEY(UserOne) REFERENCES UserProfile(UserProfileID),
constraint fk_usertwo_userprofid foreign key(USerTwo) references UserPRofile(UserProfileID)
);

-- Insert values: a connected with b
-- (a, b) and (b, a)

INSERT INTO UserConnections
VALUES(1, 2, 1, 0, NOW());
INSERT INTO UserConnections
VALUES(2, 1, 1, 0, NOW());

-- (a, c) and (c, a)

insert into UserConnections values(1, 3, 1, 0, now());
insert into UserConnections values(3, 1, 1, 0, now());

 -- (a, d) d follows a
insert into UserConnections values(1, 4, 0, 1, now());

SELECT * FROM UserConnections;

-- Create Table User Posts
-- One user can post many posts.

create table Post (
PostID      integer primary key auto_increment,
Title       varchar(250) not null,
Content      text not null,
PostedBy     integer not null,
PostedDTTM   datetime not null default now(),
constraint fk_postedby_userid foreign key(PostedBy) references UserProfile(UserProfileID)
);

insert into post(Title, Content, PostedBy)
values('SamplePost', 'Sample Post Content', 1);

select * from post;

-- Post likes Table

create table PostLike (
PostLikeID    integer primary key auto_increment,
PostID        integer not null,
LikedBy       integer not null,
ActionDTTM    datetime not null default now(),
constraint fk_postid foreign key(PostID) references Post(PostID),
constraint fk_likedby foreign key(LikedBy) references UserProfile(UserProfileID),
constraint unq_postid_likeby unique(PostID, LikedBy)
);

insert into PostLike(PostID, LikedBy)
values(1 , 2);

insert into PostLike(PostID, LikedBy)
values(1, 3);

select * from postlike;

```

```
-- Post Comment Table

create table PostComment (
PostCommentID        integer primary key auto_increment,
PostID               integer not null,
CommentForCommentID   integer,
CommentText          text not null,
CommentedBy          integer not null,
constraint fk_postid_postcomment foreign key(PostID) references Post(PostID),
constraint fk_commentedby foreign key(CommentedBy) references UserProfile(UserProfileID),
constraint fk_comment_for_comment foreign key(CommentForCommentID) references
PostComment(PostCommentID)
);

insert into PostComment(PostID, CommentText, CommentedBy)
values(1, 'good post', 2);

insert into PostComment(PostID, CommentForCommentID, CommentText, CommentedBy)
values(1, 1, 'well said', 3);

select * from PostComment;

-- retrieve the posts posted by 'a'
select * from Post
where PostedBy = (select UserProfileID from UserProfile where username = 'a');

-- retrieve the no of likes for postID - 1
select count(*) as 'Likes for post 1' from PostLike
where PostID = 1;
```

```sql
-- Problem 1) Write a query to get connection of a (solved using subqueries)
select UserName as connections_with_a from UserProfile
where UserProfileID in (
                  select UserTwo from UserConnections
     where UserOne = ( select UserProfileID from UserProfile where UserName = 'a')
     and ISConnection = 1
);

-- get connection of a (solved using join)
select UserName as connections_with_a from UserProfile as u
inner join UserConnections as uc
on (u.UserProfileID = uc.UserTwo
         and UserOne = (select UserPRofileID from UserProfile where UserName = 'a')
         and ISConnection = 1);

-- Problem 2) : Write a query to retreive all the followers of 'a'
--          Note:- followers includes connections + only followers.
select UserName from UserProfile as u
inner join UserConnections as uc
on (u.UserProfileID = uc.UserTwo
and UserOne = (select UserProfileID from UserProfile where UserName = 'a')
and (ISConnection = 1 or ISFollower = 1));

-- Problem 3) Display the total likes for the posts posted by 'a'
select count(*) as 'Likes for post by a' from PostLike
where PostID = (select UserProfileID from UserProfile where username = 'a');

-- Problem 4) Display the PostID and Likes count for the posts posted by 'a'
select PostID, count(*) as 'Likes for post by a' from PostLike
where PostID = (select UserProfileID from UserProfile where username = 'a');
```

```
-- Problem 6) Display the name of the user with maximum post likes.
-- (give name for derived table using AS)
select UserName from UserProfile as u
inner join Post as p
on (u.UserProfileID = p.PostedBy
and PostID = (select PostID from
(select PostID, max(mycount) from
(select PostID, count(PostID) as mycount from PostLike) as pi_cnt ) as pi_max));


-- Problem 7) Display the users with maximum posts.
select UserName from UserProfile as u
inner join Post as p
on (u.UserProfileID = p.PostedBy
and PostedBy = (select PostedBy from
(select PostedBy, max(mycount) from
(select Postedby, count(PostedBy) as mycount from Post) as po_cnt ) as po_max));
```