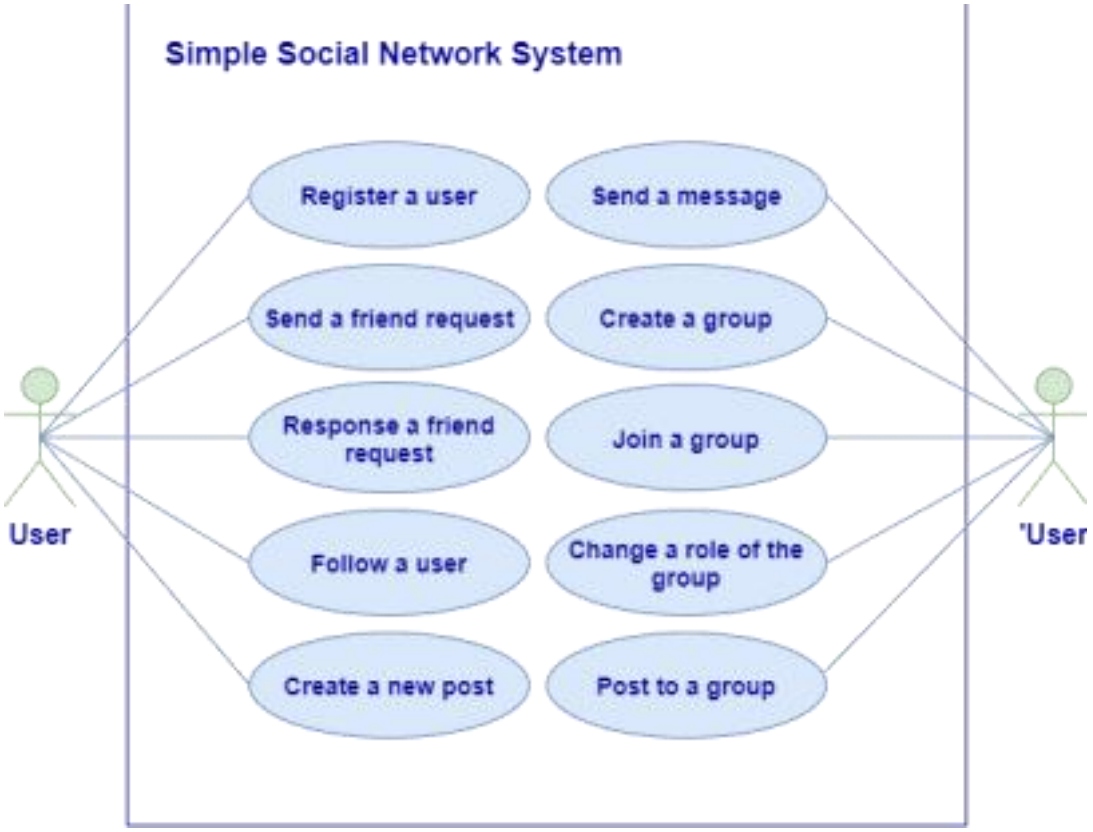
Lab02 – SQL -DDL

Objective: To study and practice the SQL commands to create database objects and manipulate data in the relational database.

Estimated Time: 3 hours

Description: Simple Social Network System (SNS)



As in the in-class practices, we have created two tables with related keys and constraints to support the use case **'Register a user', 'Send a friend request' and 'Response to a friend request'**.

This take-home lab will require you to create more stuff for the use case **'Create a new post'** and **'Send a message'**. Please consider the following instructions and do the given practices. Some items will ask you to show the SQL command while others will require you to explain your reason on your design as well as your implemented SQL command for the above system.

Reference for SQL commands: https://www.w3schools.com/sql/

Instruction:

Task 1 to 7 refer to the use case 'Create a new post'.

Task 1:

Use the given table description to create a table ***user_post***. This table can support a post on a public news feed or a post on the other user's wall. Show your CREATE TABLE command with the appropriate data types, primary key(s) and necessary constraints (foreign keys and unique columns, etc.).

Column Name	Description
id	The unique id to identify the post.
posterId	The user id to identify who create the post
profileId	The profile of the target user where the poster wants to post to (leave it null for the public post).
message	The message body.
createAt	It stores the date and time at which the post was created.
updatedAt	It stores the date and time at which the post was updated.

```
CREATE TABLE `sns`.`user_post`(
`id` BIGINT NOT NULL AUTO_INCREMENT PRIMARY KEY,
`posterId` int NOT NULL,
`profileId` int NOT NULL,
`message` VARCHAR(50) NULL,
`createAt` DATETIME not null default now(),
updatedAt DATETIME NOT NULL DEFAULT NOW() ON UPDATE NOW(),
  CONSTRAINT fk_post_poster
    FOREIGN KEY (posterId) REFERENCES user(id)
    ON DELETE CASCADE,
  CONSTRAINT fk_post_profile
    FOREIGN KEY (profileId) REFERENCES user(id)
    ON DELETE SET NULL);
```

Task 2:

From your *user_post* table, which columns you have set to be a NOT NULL constraint? List those columns and explain your reason.

Id : Id is primary key

PosterID: Post needs poster ID
ProfileID: Profile is need for post
createAt: Post always has a create date

Task 3:

Explain the foreign keys of this table. What options you have chosen **(RESTRICT, CASCADE, or SET NULL)** and the reason why you did?

Poster ID and Profile ID are foreign keys as they represent different users from user table and profile ids are useful for identifying different profiles.

For Poster ID, if the user deletes their account, then their posts should be removed. Hence, ON DELETE CASCADE is used.

If Profile is deleted, the post can still be a public post hence just ON DELETE SET NULL is used.

Task 4:

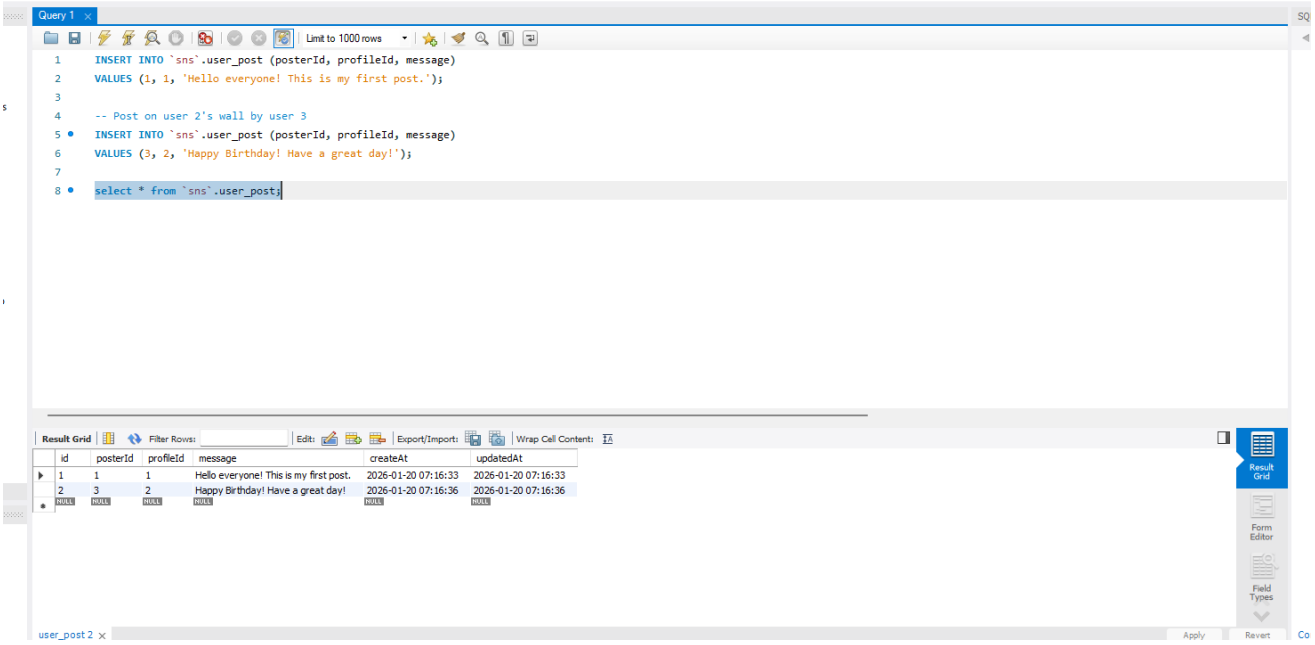
Insert two different posts from two different users. The first one is for a public post and the second is posted to the other user's wall.

INSERT INTO `sns`.user_post (posterId, profileId, message)

VALUES (1, 1, 'Hello everyone! This is my first post.');

INSERT INTO `sns`.user_post (posterId, profileId, message)

VALUES (3, 2, 'Happy Birthday! Have a great day!');



Task 5:

Suppose a user with userId = 3 wants to remove all the posts that they have ever created. Write an appropriate SQL command to delete all the posts created by the user. (both public and private posts)

delete from `sns`.user_post where posterId=3;

Task 6:

Suppose the system allows users to add a photo to the post, how do you modify the user_post table structure? [Hint: Add a new column called attachment/photos]

ALTER TABLE `sns`.user_post
ADD COLUMN attachment VARCHAR(255) NULL;

Task 7:

All posts on the social network system are monitored. Some of them can be deleted by the administrator if they violate the terms and conditions of usage (e.g. hate speech and fake news, etc.). What is the appropriate **DELETE** command to remove such violating posts (suppose that the administrator knows the id of the post they want to delete).

Delete from `sns`.user_post where id=x;

Here, we assume that X is the id of the post that they want to remove

Task 8 to 10 refer to the use case 'Send a message'.

Task 8:

Use the given table description to create a table **user_message**. This table can support sending a message to the desired user. Show your **CREATE TABLE** command with the appropriate data types, primary key(s) and necessary constraints (foreign keys and unique columns, etc.).

Column Name	Description
id	The unique id to identify the message
sourceId	The user id to identify the sender.
targetId	The user id to identify the receiver.
message	The message body.
createdAt	It stores the date and time at which the post was created. Set the default value as now().
updatedAt	It keeps the date and time at which the post was updated.

```
CREATE TABLE `sns`.user_message (
   id BIGINT AUTO_INCREMENT PRIMARY KEY,
   sourceId bigint NOT NULL,
   targetId bigint NOT NULL,
   message VARCHAR(255) NOT NULL,
   createdAt DATETIME NOT NULL DEFAULT NOW(),
   updatedAt DATETIME NOT NULL DEFAULT NOW() ON UPDATE NOW(),
   CONSTRAINT fk_message_sender
      FOREIGN KEY (sourceId) REFERENCES user(id)
      ON DELETE CASCADE,
   CONSTRAINT fk_message_receiver
      FOREIGN KEY (targetId) REFERENCES user(id)
      ON DELETE CASCADE
);
```

Task 9:

From your user_message table, are there any columns that can be NULLABLE? Explain all such columns and explain your reason.

No rows can be null. Both target and sender are needed, id is primary key, message cannot be null as it needs to be recorded and timestamps are to be recorded as good practice.

Task 10:

Insert data to the user_message table to store 2-exchange conversation messages between 2 users. List all INSERT commands.

insert into `sns`.user_message(sourceId,targetId,message) values(6,9,'Hello');
insert into `sns`.user_message(sourceId,targetId,message) values(9,6,'Hello! How may I help you?');

```
22      CONSTRAINT fk_message_receiver
23          FOREIGN KEY (targetId) REFERENCES user(id)
24          ON DELETE CASCADE
25  );
26
27  insert into `sns`.user_message(sourceId,targetId,message) values(6,9,'Hello');
28  insert into `sns`.user_message(sourceId,targetId,message) values(9,6,'Hello! How may I help you?');
29  select * from `sns`.user;
30  select * from `sns`.user_message;
```

esult Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

id	sourceId	targetId	message	createdAt	updatedAt
3	6	9	Hello	2026-01-20 07:31:16	2026-01-20 07:31:16
4	9	6	Hello! How may I help you?	2026-01-20 07:33:01	2026-01-20 07:33:01
NULL	NULL	NULL	NULL	NULL	NULL