III. Appendix - SQL Queries (separate file)

```
-- ----------------------------------------------------------------
-- SET ENGINE TO InnoDB
-- ----------------------------------------------------------------

-- SET default_storage_engine=InnoDB;

-- ----------------------------------------------------------------
-- SCHEMA & TABLES CREATION
-- ----------------------------------------------------------------

DROP SCHEMA IF EXISTS SocialBook;
CREATE SCHEMA SocialBook;

USE SocialBook;
DROP TABLE IF EXISTS Account;
CREATE TABLE Account
(
    id        INT AUTO_INCREMENT,
    firstName CHAR(50)                    NOT NULL,
    lastName  CHAR(50)                    NOT NULL,
    email     CHAR(50)                    NOT NULL UNIQUE,
    password  CHAR(50)                    NOT NULL,
    birthDate DATE                        NOT NULL,
    gender    CHAR(6)                     NOT NULL,
    joinDate  DATE DEFAULT (CURRENT_DATE) NOT NULL,
    active    BOOL DEFAULT (TRUE)         NOT NULL,
    PRIMARY KEY (id)
);

DROP TABLE IF EXISTS Post;
CREATE TABLE Post
(
    id        INT AUTO_INCREMENT,
    accountId INT                         NOT NULL,
    content   CHAR(250)                   NOT NULL,
    sponsored BOOL DEFAULT (FALSE)        NOT NULL,
    date      DATE DEFAULT (CURRENT_DATE) NOT NULL,
    PRIMARY KEY (id),
    FOREIGN KEY (accountId) REFERENCES Account (id)
```

```
        ON UPDATE CASCADE ON DELETE NO ACTION
);

DROP TABLE IF EXISTS Comment;
CREATE TABLE Comment
(
    id        INT AUTO_INCREMENT,
    postId    INT                          NOT NULL,
    accountId INT                          NOT NULL,
    content   CHAR(250)                    NOT NULL,
    date      DATE DEFAULT (CURRENT_DATE) NOT NULL,
    PRIMARY KEY (id),
    FOREIGN KEY (postId) REFERENCES Post (id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (accountId) REFERENCES Account (id)
        ON UPDATE CASCADE ON DELETE NO ACTION
);

DROP TABLE IF EXISTS Reply;
CREATE TABLE Reply
(
    id        INT AUTO_INCREMENT,
    commentId INT                          NOT NULL,
    accountId INT                          NOT NULL,
    content   CHAR(250)                    NOT NULL,
    date      DATE DEFAULT (CURRENT_DATE) NOT NULL,
    PRIMARY KEY (id),
    FOREIGN KEY (commentId) REFERENCES Comment (id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (accountId) REFERENCES Account (id)
        ON UPDATE CASCADE ON DELETE NO ACTION
);

DROP TABLE IF EXISTS SocialGroup;
CREATE TABLE SocialGroup
(
    id          INT AUTO_INCREMENT,
    name        CHAR(100) NOT NULL,
    description CHAR(250) NOT NULL,
    PRIMARY KEY (id)
);

DROP TABLE IF EXISTS Story;
```

```
CREATE TABLE Story
(
    id        INT AUTO_INCREMENT,
    accountId INT       NOT NULL,
    photoLink CHAR(250) NOT NULL,
    postDate  DATE      NOT NULL,
    PRIMARY KEY (id),
    FOREIGN KEY (accountId) REFERENCES Account (id)
        ON UPDATE CASCADE ON DELETE CASCADE
);

DROP TABLE IF EXISTS PostLikes;
CREATE TABLE PostLikes
(
    postId    INT NOT NULL,
    accountId INT NOT NULL,
    FOREIGN KEY (postId) REFERENCES Post (id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (accountId) REFERENCES Account (id)
        ON UPDATE CASCADE ON DELETE NO ACTION,
    PRIMARY KEY (postId, accountId)
);

DROP TABLE IF EXISTS PostShares;
CREATE TABLE PostShares
(
    postId    INT NOT NULL,
    accountId INT NOT NULL,
    FOREIGN KEY (postId) REFERENCES Post (id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (accountId) REFERENCES Account (id)
        ON UPDATE CASCADE ON DELETE NO ACTION,
    PRIMARY KEY (postId, accountId)
);

DROP TABLE IF EXISTS CommentLikes;
CREATE TABLE CommentLikes
(
    commentId INT NOT NULL,
    accountId INT NOT NULL,
    FOREIGN KEY (commentId) REFERENCES Comment (id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (accountId) REFERENCES Account (id)
```

```
        ON UPDATE CASCADE ON DELETE NO ACTION,
    PRIMARY KEY (commentId, accountId)
);

DROP TABLE IF EXISTS ReplyLikes;
CREATE TABLE ReplyLikes
(
    replyId   INT NOT NULL,
    accountId INT NOT NULL,
    FOREIGN KEY (replyId) REFERENCES Reply (id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (accountId) REFERENCES Account (id)
        ON UPDATE CASCADE ON DELETE NO ACTION,
    PRIMARY KEY (replyId, accountId)
);

DROP TABLE IF EXISTS Friendship;
CREATE TABLE Friendship
(
    receiverAccountId INT                  NOT NULL,
    senderAccountId   INT                  NOT NULL,
    pending           BOOL DEFAULT (TRUE)  NOT NULL,
    denied            BOOL DEFAULT (FALSE) NOT NULL,
    FOREIGN KEY (receiverAccountId) REFERENCES Account (id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (senderAccountId) REFERENCES Account (id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    PRIMARY KEY (receiverAccountId, senderAccountId)
);

DROP TABLE IF EXISTS GroupPosts;
CREATE TABLE GroupPosts
(
    socialGroupId INT NOT NULL,
    postId        INT NOT NULL,
    FOREIGN KEY (socialGroupId) REFERENCES SocialGroup (id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (postId) REFERENCES Post (id)
        ON UPDATE CASCADE ON DELETE CASCADE
);

DROP TABLE IF EXISTS GroupParticipants;
CREATE TABLE GroupParticipants
```

```
(
    socialGroupId INT NOT NULL,
    accountId     INT NOT NULL,
    FOREIGN KEY (socialGroupId) REFERENCES SocialGroup (id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    FOREIGN KEY (accountId) REFERENCES Account (id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    PRIMARY KEY (socialGroupId, accountId)
);

-- ----------------------------------------------------------------
-- ACCOUNT POPULATION
-- ----------------------------------------------------------------

USE SocialBook;

INSERT INTO Account (firstName, lastName, email, password, birthDate,
gender)
VALUES ('Paul', 'Panaitescu', 'paul000@gmail.com', 'pass12',
'1996-10-7', 'male'),
       ('Constantin', 'Tarau', 'cons000@gmail.com', 'pass123',
'1998-09-15', 'male'),
       ('Marius', 'Munteanu', 'mari000@gmail.com', 'pass1234',
'1998-07-31', 'male'),
       ('Gianluigi', 'Buffon', 'gigi000@gmail.com', 'gigi',
'1978-01-28', 'male'),
       ('Andrea', 'Pirlo', 'andr000@gmail.com', 'pass', '1995-01-17',
'male'),
       ('Alessandro', 'Nesta', 'ales000@gmail.com', 'pass',
'1976-03-19', 'male'),
       ('Carlo', 'Ancelotti', 'carl000@gmail.com', 'pass',
'1959-06-10', 'male'),
       ('Alyssa', 'Borunda', 'AlyssaCantamessa@armyspy.com',
'daimohGh0loo', '1977-01-25', 'female'),
       ('Paolo', 'Maldini', 'paol000@gmail.com', 'pass',
'1968-06-26', 'male'),
       ('Camila', 'Giorgi', 'cami000@gmail.com', 'pass',
'1991-12-30', 'female');

-- ----------------------------------------------------------------
-- POST POPULATION
-- ----------------------------------------------------------------
```

```
USE SocialBook;

INSERT INTO Post (accountId, content, sponsored, date)
VALUES (1, 'Post number 1 - Paul', 1, '2020-05-01'),
       (2, 'Post number 1 - Constantin', 1, '2020-05-02'),
       (3, 'Post number 1 - Marius', 1, '2020-05-02'),
       (1, 'Post number 2 - Paul', 1, '2020-05-05'),
       (1, 'Post number 3 - Paul', 1, '2020-05-18'),
       (1, 'Post number 4 - Paul', 1, '2020-05-30'),
       (2, 'Post number 2 - Constantin', 1, '2020-05-05'),
       (2, 'Post number 3 - Constantin', 1, '2020-05-30'),
       (3, 'Post number 2 - Marius', 1, '2020-05-31'),
       (4, 'Post number 1 - Gianluigi', 1, '2020-05-10'),
       (5, 'Post number 1 - Andrea Pirlo', 1, '2020-05-15'),
       (10, 'Post number 1 - Camila Giorgi', 1, '2020-05-16'),
       (10, 'Post number 2 - Camila Giorgi', 1, '2020-05-17'),
       (10, 'Post number 3 - Camila Giorgi', 1, '2020-05-25'),
       (7, 'Post number 1 - Carlo Ancelotti', 1, '2020-05-05');

-- -----------------------------------------------------------------
-- COMMENT POPULATION
-- -----------------------------------------------------------------

USE SocialBook;

INSERT INTO Comment (postId, accountId, content, date)
VALUES (1, 2, 'C - Comment 1', '2020-05-01'),
       (1, 3, 'M - Comment 1', '2020-05-01'),
       (1, 2, 'C - Comment 2', '2020-05-02'),
       (1, 3, 'M - Comment 2', '2020-05-02'),
       (1, 1, 'P - Comment 1', '2020-05-02'),
       (1, 3, 'M - Comment 3', '2020-05-03'),
       (1, 3, 'M - Comment 4', '2020-05-03'),
       (1, 2, 'C - Comment 3', '2020-05-03'),
       (1, 1, 'P - Comment 2', '2020-05-03'),
       (1, 1, 'P - Comment 3', '2020-05-04'),
       (3, 1, 'P - Comment 1', '2020-05-02'),
       (3, 2, 'C - Comment 1', '2020-05-02'),
       (3, 2, 'C - Comment 2', '2020-05-02'),
       (3, 1, 'P - Comment 2', '2020-05-02'),
       (2, 3, 'M - Comment 1', '2020-05-02'),
       (2, 1, 'P - Comment 1', '2020-05-02'),
       (2, 2, 'C - Comment 1', '2020-05-02'),
       (10, 10, 'Camila - Comment 1', '2020-05-10'),
       (10, 7, 'Carlo - Comment 1', '2020-05-10'),
       (10, 10, 'Camila - Comment 2', '2020-05-10'),
       (14, 6, 'Alessandro - Comment 1', '2020-05-25'),
       (14, 8, 'Alyssa - Comment 1', '2020-05-25'),
       (14, 9, 'Paolo - Comment 1', '2020-05-25'),
       (14, 10, 'Camila - Comment 1', '2020-05-25'),
       (15, 8, 'Alyssa - Comment 1', '2020-05-15');

-- ----------------------------------------------------------------
-- REPLY POPULATION
-- ----------------------------------------------------------------

USE SocialBook;

INSERT INTO Reply (commentId, accountId, content, date)
VALUES (1, 1, 'P - Reply 1', '2020-05-01'),
       (1, 2, 'C - Reply 1', '2020-05-01'),
       (1, 3, 'M - Reply 1', '2020-05-01'),
       (1, 1, 'P - Reply 2', '2020-05-01'),
       (4, 1, 'P - Reply 1', '2020-05-02'),
       (4, 2, 'C - Reply 1', '2020-05-02'),
       (4, 1, 'P - Reply 2', '2020-05-03'),
       (4, 2, 'C - Reply 2', '2020-05-03'),
       (11, 3, 'M - Reply 1', '2020-05-02'),
       (1, 2, 'C - Reply 1', '2020-05-03'),
       (1, 3, 'M - Reply 2', '2020-05-03'),
       (1, 2, 'C - Reply 2', '2020-05-04'),
       (1, 1, 'P - Reply 1', '2020-05-04'),
       (8, 6, 'Alessandro - Reply 1', '2020-05-10'),
       (8, 7, 'Carlo - Reply 1', '2020-05-13'),
       (8, 6, 'Alessandro - Reply 2', '2020-05-15'),
       (8, 10, 'Camila - Reply 1', '2020-05-16'),
       (8, 10, 'Camila - Reply 2', '2020-05-16'),
       (23, 8, 'Alyssa - Reply 1', '2020-05-25'),
       (23, 9, 'Paolo - Reply 1', '2020-05-26'),
       (23, 8, 'Alyssa - Reply 2', '2020-05-28'),
       (19, 6, 'Alessandro - Reply 1', '2020-05-10'),
       (19, 6, 'Alessandro - Reply 2', '2020-05-20'),
       (25, 8, 'Alyssa - Reply 1', '2020-05-15'),
       (21, 4, 'Gianluigi - Reply 1', '2020-05-26');
```

```
-- -------------------------------------------------------------------
-- SOCIAL GROUP POPULATION
-- -------------------------------------------------------------------

USE SocialBook;

INSERT INTO SocialGroup (name, description)
VALUES ('Databases - Final Project', 'Chat for Final Project in
Databases'),
       ('Coppa Campioni d''Italia', 'Chat Group for Serie A
Championship'),
       ('Software Development - TOP UP', 'Chat Group for Software
Development in TOP UP'),
       ('A.C. Milan', 'Professional Football club in Milan');

-- -------------------------------------------------------------------
-- STORY POPULATION
-- -------------------------------------------------------------------

USE SocialBook;

INSERT INTO Story (accountId, photoLink, postDate)
VALUES (1, 'Link for Story 2', '2020-05-01'),
       (1, 'Link for Story 2', '2020-05-15'),
       (2, 'Link for Story 1', '2020-05-01'),
       (2, 'Link for Story 2', '2020-05-05'),
       (2, 'Link for Story 3', '2020-05-20'),
       (1, 'Link for Story 1', '2020-05-31'),
       (4, 'Link for Story 1', '2020-05-01'),
       (5, 'Link for Story 1', '2020-05-03'),
       (8, 'Link for Story 1', '2020-05-01'),
       (8, 'Link for Story 2', '2020-05-02'),
       (8, 'Link for Story 3', '2020-05-03'),
       (8, 'Link for Story 4', '2020-05-04'),
       (8, 'Link for Story 5', '2020-05-05'),
       (9, 'Link for Story 1', '2020-05-27'),
       (10, 'Link for Story 1', '2020-05-10'),
       (10, 'Link for Story 2', '2020-05-12'),
       (10, 'Link for Story 3', '2020-05-14'),
       (10, 'Link for Story 4', '2020-05-16'),
       (10, 'Link for Story 5', '2020-05-16'),
       (10, 'Link for Story 6', '2020-05-30');

-- -----------------------------------------------------------------
-- POST LIKES POPULATION
-- -----------------------------------------------------------------

USE SocialBook;

INSERT INTO PostLikes (postId, accountId)
VALUES (2, 1),
       (3, 1),
       (9, 1),
       (10, 1),
       (11, 1),
       (15, 1),
       (1, 2),
       (3, 2),
       (9, 2),
       (4, 2),
       (12, 2),
       (15, 2),
       (7, 3),
       (8, 3),
       (5, 3),
       (6, 3),
       (10, 4),
       (11, 4),
       (1, 5),
       (2, 5),
       (3, 5),
       (15, 9),
       (11, 10),
       (10, 10),
       (15, 10);

-- -----------------------------------------------------------------
-- POST SHARES POPULATION
-- -----------------------------------------------------------------

USE SocialBook;
```

```
INSERT INTO PostShares (postId, accountId)
VALUES (2, 1),
       (3, 1),
       (1, 2),
       (3, 2),
       (4, 3),
       (7, 3),
       (11, 4),
       (15, 10),
       (15, 4),
       (15, 7);


-- ----------------------------------------------------------------
-- COMMENT LIKES POPULATION
-- ----------------------------------------------------------------

USE SocialBook;

INSERT INTO CommentLikes (commentId, accountId)
VALUES (1, 1),
       (7, 1),
       (8, 1),
       (10, 2),
       (14, 2),
       (15, 2),
       (5, 3),
       (16, 3),
       (17, 3),
       (19, 10),
       (22, 6),
       (25, 10),
       (22, 10),
       (22, 7),
       (21, 10);
```

```
-- ----------------------------------------------------------------
-- REPLY LIKES POPULATION
-- ----------------------------------------------------------------

USE SocialBook;

INSERT INTO ReplyLikes (replyId, accountId)
VALUES (2, 1),
       (3, 1),
       (10, 1),
       (1, 2),
       (3, 2),
       (11, 2),
       (1, 3),
       (2, 3),
       (12, 3),
       (14, 4),
       (16, 5),
       (25, 5),
       (18, 8),
       (19, 10),
       (24, 10);

-- ----------------------------------------------------------------
-- FRIENDSHIP POPULATION
-- ----------------------------------------------------------------

USE SocialBook;

INSERT INTO Friendship (receiverAccountId, senderAccountId, pending,
denied)
VALUES (2, 1, 0, 0),
       (3, 1, 0, 0),
       (1, 2, 0, 1),
       (3, 2, 0, 0),
       (1, 3, 0, 0),
       (2, 3, 1, 0),
       (7, 4, 0, 1),
       (10, 4, 1, 0),
       (7, 6, 0, 0),
       (8, 10, 0, 1);
```

```
-- -----------------------------------------------------------------
-- GROUP POSTS POPULATION
-- -----------------------------------------------------------------

USE SocialBook;

INSERT INTO GroupPosts (socialGroupId, postId)
VALUES (1, 1),
       (1, 2),
       (1, 3),
       (3, 4),
       (3, 5),
       (1, 6),
       (3, 7),
       (3, 8),
       (3, 9),
       (2, 10),
       (2, 11),
       (4, 12),
       (4, 13),
       (2, 14),
       (4, 15);

-- -----------------------------------------------------------------
-- GROUP PARTICIPANTS POPULATION
-- -----------------------------------------------------------------

USE SocialBook;

INSERT INTO GroupParticipants (socialGroupId, accountId)
VALUES (1, 1),
       (1, 2),
       (1, 3),
       (3, 1),
       (3, 2),
       (3, 3),
       (2, 4),
       (2, 5),
       (2, 6),
       (2, 7),
       (2, 8),
       (2, 9),
       (2, 10),
       (4, 6),
       (4, 7),
       (4, 8),
       (4, 10);

-- ---------------------------------------------------------------------
-- WELCOME POST PROCEDURE
-- ---------------------------------------------------------------------

USE SocialBook;

DELIMITER $$

DROP PROCEDURE IF EXISTS create_welcome_post $$
CREATE PROCEDURE create_welcome_post(accId INT, accFirstName
CHAR(50))
BEGIN
    INSERT INTO Post(accountId, content, date)
    VALUES (accId, CONCAT(accFirstName, ' joined SocialBook!'),
current_date());
END $$

DROP TRIGGER IF EXISTS welcome_post $$
CREATE TRIGGER welcome_post
    AFTER INSERT
    ON Account
    FOR EACH ROW
BEGIN
    CALL create_welcome_post(
            NEW.id,
            NEW.firstName
        );
END $$

DELIMITER ;

-- ---------------------------------------------------------------------
-- DISABLE ACCOUNT PROCEDURE
-- ---------------------------------------------------------------------

USE SocialBook;

DELIMITER $$
```

```
CREATE PROCEDURE disable_account(accountId INT)
BEGIN
    UPDATE Account SET active = 0 WHERE Account.id = accountId;
END $$

DELIMITER ;

-- ----------------------------------------------------------------
-- USER ROLES AND PRIVILEGES
-- ----------------------------------------------------------------

USE SocialBook;

DROP ROLE IF EXISTS 'role_administrator', 'role_create', 'role_read',
'role_update', 'role_delete';
CREATE ROLE 'role_administrator', 'role_create', 'role_read',
'role_update', 'role_delete';

GRANT ALL ON *.* TO 'role_administrator' WITH GRANT OPTION;
GRANT INSERT ON *.* TO 'role_create';
GRANT SELECT ON *.* TO 'role_read';
GRANT UPDATE ON *.* TO 'role_update';
GRANT DELETE ON *.* TO 'role_delete';

DROP USER IF EXISTS 'user_admin'@'localhost';
DROP USER IF EXISTS 'user_developer'@'localhost';
DROP USER IF EXISTS 'user_basic'@'localhost';

CREATE USER 'user_admin'@'localhost' IDENTIFIED BY 'password';
CREATE USER 'user_developer'@'localhost' IDENTIFIED BY 'password';
CREATE USER 'user_basic'@'localhost' IDENTIFIED BY 'password';

GRANT 'role_administrator' TO 'user_admin'@'localhost';
GRANT 'role_create', 'role_read', 'role_update', 'role_delete' TO
'user_developer'@'localhost';
GRANT 'role_create', 'role_read' TO 'user_basic'@'localhost';

SET DEFAULT ROLE ALL TO
  'user_admin'@'localhost',
  'user_developer'@'localhost',
  'user_basic'@'localhost';
SELECT *
```

```
FROM mysql.user;
-- ----------------------------------------------------------------
-- USER GRANTS
-- ----------------------------------------------------------------

SELECT CURRENT_USER();
SELECT CURRENT_ROLE();
SHOW GRANTS FOR CURRENT_USER();

-- ----------------------------------------------------------------
-- VIEWS (for BASIC USER, DEVELOPER, ADMIN)
-- ----------------------------------------------------------------

USE SocialBook;

DROP VIEW IF EXISTS Basic_User_View_Account;
CREATE VIEW Basic_User_View_Account AS
SELECT id, firstName, lastName, gender, joinDate
FROM Account;

DROP VIEW IF EXISTS Developer_User_View_Account;
CREATE VIEW Developer_User_View_Account AS
SELECT id,
       firstName,
       lastName,
       email,
       birthDate,
       gender,
       joinDate,
       active
FROM Account;

DROP VIEW IF EXISTS Admin_User_View;
CREATE VIEW Admin_User_View AS
SELECT User, Select_priv, Insert_priv, Update_priv, Delete_priv,
Create_priv
FROM mysql.user
WHERE (User LIKE 'user_%')
   OR (User LIKE 'role_%');
```

```
-- -----------------------------------------------------------------
-- FLUSH PRIVILEGES
-- -----------------------------------------------------------------

FLUSH PRIVILEGES;

-- -----------------------------------------------------------------
-- BIRTHDAY POST EVENT SCHEDULER
-- -----------------------------------------------------------------

USE SocialBook;

DROP EVENT IF EXISTS birthday_schedule;
DELIMITER $$

CREATE EVENT birthday_schedule
    ON SCHEDULE EVERY 1 DAY
        STARTS CURRENT_DATE()
    DO
    BEGIN
        INSERT INTO Post(accountId, content, date)
        SELECT id, 'Happy Birthday!', NOW()
        FROM Account
        WHERE active = 1
          AND CURRENT_DATE() = DATE_ADD(birthDate, INTERVAL
YEAR(CURRENT_DATE()) - YEAR(birthDate) YEAR);
    END $$

DELIMITER ;

-- -----------------------------------------------------------------
-- RECENT POSTS TEMPORARY TABLE
-- -----------------------------------------------------------------

USE SocialBook;

DROP FUNCTION IF EXISTS get_n_likes;
DROP FUNCTION IF EXISTS get_n_shares;
DROP FUNCTION IF EXISTS get_n_comments;
DROP PROCEDURE IF EXISTS get_posts_of;
DROP TABLE IF EXISTS RecentPosts;
```

```
-- SET GLOBAL log_bin_trust_function_creators = 1;

DELIMITER $$

CREATE FUNCTION get_n_likes(postId INT)
    RETURNS INT
BEGIN
    DECLARE temp INT;
    SELECT COUNT(*) INTO temp FROM postlikes WHERE postlikes.postId =
postId;
    RETURN temp;
END $$

CREATE FUNCTION get_n_comments(postId INT)
    RETURNS INT
BEGIN
    DECLARE temp INT;
    SELECT COUNT(*) INTO temp FROM comment AS c WHERE c.postId =
postId;
    RETURN temp;
END $$

CREATE FUNCTION get_n_shares(postId INT)
    RETURNS INT
BEGIN
    DECLARE temp INT;
    SELECT COUNT(*) INTO temp FROM postshares AS ps WHERE ps.postId =
postId;
    RETURN temp;
END $$

CREATE PROCEDURE get_posts_of(accId INT)
BEGIN

    CREATE TEMPORARY TABLE IF NOT EXISTS RecentPosts
    (
        accountId INT NOT NULL,
        postId    INT NOT NULL,
        likes     INT,
        comments  INT,
        shares    INT,
        firstName CHAR(50),
        lastName  CHAR(50),
        content   CHAR(250),
        sponsored BOOL,
        date      DATE,
        PRIMARY KEY (accountId, postId)
    );

    IF (SELECT COUNT(*) FROM RecentPosts WHERE accountId = accId) = 0
THEN
        INSERT INTO RecentPosts
        SELECT a.id,
               p.id,
               get_n_likes(p.id),
               get_n_comments(p.id),
               get_n_shares(p.id),
               a.firstName,
               a.lastName,
               p.content,
               p.sponsored,
               p.date
        FROM Post as p
                 INNER JOIN Account as a ON a.id = p.accountId
        WHERE accountId = accId;
    END IF;

    SELECT * FROM RecentPosts WHERE accountId = accId;
END $$

DELIMITER ;

-- ----------------------------------------------------------------
-- FRIEND REQUEST VALIDATION FUNCTIONS & TRANSACTION
-- ----------------------------------------------------------------

USE SocialBook;

DELIMITER $$

DROP TRIGGER IF EXISTS validate_request $$
DROP PROCEDURE IF EXISTS add_friend_request $$

CREATE TRIGGER validate_request
    BEFORE INSERT
    ON Friendship
    FOR EACH ROW
BEGIN
    DECLARE error_msg varchar(255);
    IF (SELECT COUNT(*)
        FROM Friendship
        WHERE senderAccountId = NEW.receiverAccountId
          AND receiverAccountId = NEW.senderAccountId) != 0 THEN
        SET error_msg =
                'FriendshipError: Trying to insert a friendship where
senderAccountId and receiverAccountId were found in reversed places
in the table';
        SIGNAL SQLSTATE '45000' SET message_text = error_msg;
    ELSEIF NEW.receiverAccountId = NEW.senderAccountId THEN
        SET error_msg =
                'FriendshipError: Trying to insert a friendship where
senderAccountId and receiverAccountId are the same';
        SIGNAL SQLSTATE '45000' SET message_text = error_msg;
    ELSEIF (SELECT COUNT(*)
            FROM Friendship
            WHERE senderAccountId = NEW.senderAccountId
              AND receiverAccountId = NEW.receiverAccountId) != 0
THEN
        SET error_msg =
                'FriendshipError: Trying to insert a friendship where
senderAccountId and receiverAccountId already exist in the table';
        SIGNAL SQLSTATE '45000' SET message_text = error_msg;
    END IF;
END $$

CREATE PROCEDURE add_friend_request(senderId INT, receiverId INT)
BEGIN
    DECLARE rollback BOOL DEFAULT 0;
    DECLARE CONTINUE HANDLER FOR SQLEXCEPTION
        SET rollback = 1;

    INSERT INTO Friendship VALUES (receiverId, senderId, TRUE,
FALSE);

    IF rollback THEN
        ROLLBACK;
    ELSE
        COMMIT;
    END IF;
END $$

DELIMITER ;

CALL add_friend_request(1, 1);
CALL add_friend_request(1, 2);
CALL add_friend_request(1, 2);
CALL add_friend_request(2, 1);

-- ----------------------------------------------------------------
-- ACCOUNT NAME INDEX
-- -----------------------------------------------------------------

USE SocialBook;

-- DROP INDEX account_name ON Account;

CREATE INDEX account_name ON Account (firstName, lastName);

SELECT *
FROM Account
         USE INDEX (account_name)
WHERE firstName LIKE 'm%'
   OR lastName LIKE 'm%';

-- ----------------------------------------------------------------
--                         --- END ---
-- ----------------------------------------------------------------
```