DELIMITER ;;

CREATE DEFINER=`root`@`localhost` FUNCTION `darAdmin`(idUsuario INT) RETURNS int(11)

BEGIN

UPDATE usuarios\_registrados SET es\_admin = 1 WHERE id = idUsuario;

RETURN ROW\_COUNT();

END ;;

DELIMITER ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` FUNCTION `quitarAdmin`(idUsuario INT) RETURNS int(11)

BEGIN

UPDATE usuarios\_registrados SET es\_admin = 0 WHERE id = idUsuario;

RETURN ROW\_COUNT();

END ;;

DELIMITER ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `EliminarUsuario`(

IN userId INT

)

BEGIN

DELETE FROM usuarios\_registrados WHERE id = userId;

END ;;

DELIMITER ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `sp\_agregar\_valoracion`(IN p\_articulo\_id INT, IN p\_usuario\_id INT, IN p\_valoracion ENUM('like', 'dislike'))

BEGIN

DECLARE valoracionExistente ENUM('like', 'dislike');

-- Verifica si ya existe una valoración

SELECT valoracion INTO valoracionExistente

FROM valoraciones\_articulos

WHERE articulo\_id = p\_articulo\_id AND usuario\_id = p\_usuario\_id;

IF valoracionExistente IS NOT NULL THEN

-- Actualiza la valoración existente

UPDATE valoraciones\_articulos

SET valoracion = p\_valoracion

WHERE articulo\_id = p\_articulo\_id AND usuario\_id = p\_usuario\_id;

ELSE

-- Inserta una nueva valoración

INSERT INTO valoraciones\_articulos (articulo\_id, usuario\_id, valoracion)

VALUES (p\_articulo\_id, p\_usuario\_id, p\_valoracion);

END IF;

-- Llama al trigger para actualizar conteos

CALL update\_likes\_dislikes();

END ;;

DELIMITER ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` TRIGGER log\_user\_activity

AFTER INSERT ON comentarios

FOR EACH ROW

BEGIN

INSERT INTO user\_activity\_log (usuario, actividad, fecha\_hora)

VALUES (NEW.usuario, 'Nuevo comentario', NOW());

END ;;

DELIMITER ;